JGrass 2.0

User Manual

The HydroloGIS Team
Andrea Antonello & Silvia Franceschi

In collaboration with CUDAM
Riccardo Rigon & Alfonso Vitti

UNIVERSITÀ DEGLI STUDI DI TRENTO
Dipartimento di Ingegneria Civile
e Ambientale

June 2006

ISBN 10: 88-8443-145-X
ISBN 13: 978-88-8443-145-5

Andrea Antonello
andrea.antonello@hydrologis.com

Silvia Franceschi
silvia.franceschi@hydrologis.com

Riccardo Rigon
riccardo.rigon@ing.unitn.it

Alfonso Vitti
alfonso.vitti@ing.unitn.it

HydroloGIS
www.hydrologis.com

Translation by Joseph E. Tomasi

Thanks to Vanna Antonello, the patient “guinea-pig” user

Readme

This ebook was written by Silvia Franceschi, Andrea Antonello ,Riccardo Rigon and Alfonso Vitti(HydroloGIS - Università degli Studi di Trento, Department of Civil and Environmental Engineering).

It is distributed according to the CREATIVE COMMONS deed: Attribution-NoDerivs 2.5 According to this license type you are free to:

- copy, distribute, display, and perform the work
- to make commercial use of the work

Under the following conditions:

- Attribution. You must attribute the work in the manner speciied by the author or licensor.
- No Derivative Works. You may not alter, transform, or build upon this work.
- For any reuse or distribution, you must make clear to others the license terms of this work.

Any of these conditions can be waived if you get permission from the copyright holder.

Your fair use and other rights are in no way affected by the above. This is a human-readable summary of the Legal Code (the full license) that can be consulted at:
http://creativecommons.org/licenses/by-nd/2.5/legalcode

Contents

1 Introduction **1**

1.1 The JGrass GIS 1

1.2 Future Developments 3

1.3 Extra Information 3

2 An Introduction to GIS **5**

2.1 Introduction 5

2.1.1 Definitions 5

2.1.2 GIS and LIS 6

2.2 Data Model 6

2.3 Types of Geographic Data 7

2.4 Functionality 7

2.5 Glossary 8

3 Installation **15**

3.1 System Requirements 15

3.2 Software Installation 15

3.2.1 Jar File Installation 15

3.2.2 The Installation Process 16

3.3 External Programs 21

3.3.1 The R Packages Used 22

3.4 Structure of the Installation Directories 23

3.5 Data Structure 24

4 First Start-Up of the Program **29**

4.1 Introduction 29

4.2 The Graphic Interface 30

4.2.1 The Toolbar 30

4.2.2 The Main Menu . . . 31
4.2.3 The Workspace . . . 31
4.2.4 The Monitor Toolbar . . . 33
4.2.5 The Menu Bar . . . 35
4.3 Customization . . . 43
4.4 Display Properties . . . 47

5 First Steps with JGrass **49**
5.1 The Spearfish Test Data Set . . . 49
5.2 Region Parameters . . . 49
5.2.1 Interactive region tool . . . 50
5.3 Viewing the Data . . . 52
5.3.1 Raster Data . . . 52
5.3.2 Vector Data . . . 59
5.4 Map Query . . . 64
5.4.1 Querying Raster Maps . . . 64
5.4.2 Querying Vector Maps . . . 65
5.5 Plotting Map Profiles . . . 66
5.6 Handling Raster Map . . . 69
5.6.1 Copying a Map . . . 69
5.6.2 Renaming a Map . . . 72
5.6.3 Removing a Map . . . 73
5.7 Creating a Colormap for a Raster Map . . . 74
5.8 Calculations with Raster Maps . . . 78
5.9 Creating a Category Table . . . 80
5.10 Editing Raster Map Values . . . 83
5.10.1 Editing Raster Maps Manually . . . 83
5.10.2 Editing Raster Maps with the Use of Polygons . . . 84

6 Creation of a New Workspace **87**
6.1 Creation of a Location . . . 87
6.1.1 Loading an Existing Location . . . 93
6.2 Choice of the Reference System and Cartographic Projection . . . 94
6.3 Importing Raster Data . . . 97
6.3.1 Importing Raster Maps . . . 97

6.3.2 Importing Vector Maps . . . 105
6.4 Data Re-Projection . . . 105

7 Special Operations **109**
7.1 Creating Contour Lines from a DEM . . . 109
7.2 Creating a DEM from Contour Lines . . . 110

8 Environmental Databases **113**
8.1 Installation and Usage of a Database Client . . . 113
8.1.1 Creation of the Remote Database Connections . . . 114
8.1.2 Connection and Data Visualization . . . 116
8.2 How to Interface JGrass to the Database . . . 118
8.2.1 The Interface for a Point Database . . . 118
8.2.2 The Interface for a Spatial Database . . . 124

9 Use of the Console **129**
9.1 Execution of a JGrass Command from the Console . . . 129
9.2 Scripting . . . 131
9.3 Execution of GRASS Commands from the Console . . . 131
9.4 Index of GRASS Commands . . . 132

10 Statistical Processing **143**
10.1 Statistical Processing of Point Maps . . . 143
10.2 Statistical Report of the Data Set . . . 147
10.3 The Probability Density Function . . . 148
10.4 The Cumulative Distribution Function . . . 149
10.5 Geostatistical Interpolation with the Kriging Interpolator . . . 150

11 Printing **157**

12 Uninstalling **161**

1 Introduction

We have been asked to write a *JGrass* manual for quite some time, and now we have finally decided to do so.

The manual that we now present to you tries to follow the logical steps applied when using *JGrass*, starting with a presentation of the programme and the main algorithms that are implemented, then passing to the definition of particular environmental parameters (in particular hydrological and geomorphological ones). This manual will give you a general overview of the capacities of *JGrass*, it will also give you some hints and ideas on how to solve certain spatial problems.

In the HTML version of this manual [*http://geoserver.ing.unitn.it/jgrass*] there are two ways to proceed: the pages are ordered in a linear fashion. At the beginning and at the end of each page there are four directions to choose from. The left arrow takes you back to the previous page (which not necessarily is the last page viewed) while the right arrow brings you to the next page of the manual. Clicking on the link to the index brings you to the contents page of the manual, with a brief description of each chapter. Finally, the 'A-Z' link brings you to an analogical index of all the *JGrass* modules.

1.1 The JGrass GIS

The research centre for the development of the algorithms implemented in *JGrass* is the Department of Civil and Environmental Engineering of the University of Trento (Italy) and CUDAM (University Centre for the Defence of Mountain Environments), also at the University of Trento. The development centre is HydroloGIS of Bolzano (Italy), although there are other developers at CUDAM in Trento and at ICENS in Kingston (Jamaica)

The *JGrass* starting point is GRASS GIS, which currently represents the most important Open Source project in the GIS field. GRASS is slowly evolving towards an attractive solution for the commercial and productive sectors. The biggest obstacle to the commercial blooming of GRASS is without doubt its versatility problem. Many professionals use Windows and Mac-OS as the preferred operating systems, and many businesses do not provide the possibility of using Linux (which is the operating system upon which GRASS development is based).

JGrass, developed to simplify the use of GRASS on the various platforms, has set the following objectives:

- versatility - the possibility to run GRASS on Windows, Unix/Linux and Mac-OS indifferently

(write once, run everywhere)

- the possibility for anyone, by means of the framework that has been created, to develop with ease additional modules, particular functions, and application oriented interfaces by means of XML scripting

- scripting - the possibility to use a Java-based scripting language in order to automatize certain processes (programmer and scientific-work oriented)

- simplicity of use - menu bars, tool bars and icons aimed at simplifying the interface (end-user friendly)

Over the last number of years various attempts have been made to join the strengths of Java and GRASS, the first of these was the wrapping of the GRASS libraries in Java by Alexandre Sorokine. Shortly thereafter, at the Centre for Environmental and Nuclear Sciences of the University of the West Indies (Jamaica), John Preston began the development of a Java/Swing application for the visualization of geochemical maps.

At the beginning of 2003 Dr.Rigon, of the Department of Civil and Environmental Engineering and CUDAM of the University of Trento, decided to start a new project on the shoulder of the the Jamaican, and he supplyed funds and a developer, and coordinated with the research group at ITC, which includes Markus Neteler. In this way began the development of *JGrass*.

He also disclosed the code of the Horton Machine, a large set of Tools for terrain analysis which are one of the largest application of JGrass 2.0.

At the moment *JGrass* is an application that can be run locally, although the future implementation of remote running is planned. It is a Java/Swing application based on an MDI (multiple document interface) and provides standardized ways of interaction between the interface and the GRASS kernel.

Although we are trying to keep *JGrass*, as much as possible, in "pure Java", some parts are still written in the native language of GRASS by means of Sun's API JNI technology.

Within the *JGrass* structure there are a 2D/3D display monitor based on OpenGl technology (GL4java, Jogl) and a console, from where it is possible to launch applications and all the original GRASS commands. There are also toolbars and a menu bar that can be used instead of the console.

In this manual a project that makes use raster map visualizations, vector files, sites and even shapefiles (ESRI), 2D and 3D visualizations, and some printing and exportation utilities will be presented.

1.2 Future Developments

JGrass has evolved greatly over the last two years thanks to the experience acquired by the developers and to the interest of a great number of people. It can propose itself in the near future as a complete and easy-to-use GIS for environmental analyses, comparable to the better-known ArcView.

The development of a complete vector environment is expected shortly. It will include vector editing and it could be used as a viewing instrument for remote maps, in other words it would be a complete WebGIS for analysis and calculations of maps stored on a central database.

Another area under development is the possibility of processing large quantities of data. LIDAR (Laser Scanner) surveys are becoming ever more common, they allow for very precise representations of a land area and the preparation of digital elevation models with resolutions as fine as $1m \times 1m$. There is, therefore, a substantial amount of data that must be visualized and processed in order to carry out the required environmental analysis. *JGrass* will be enhanced with an innovative system for data visualization (ROAM), similar to that used at Livermore Laboratories of Berkeley, as well as an efficient data mapping system for the number crunching routines.

1.3 Extra Information

We have been working on the *JGrass* Project for a number of years, and a complete series of tools has been activated for both users and programmers. These tools are:

- mailing list: for users and programmers, addresses and registration instructions can be found at *http://www.hydrologis.com/jgrassml.html* in the *Tools* section
- bug tracker: to report bugs in the software, to send a message all you have to do is go to *https://sourceforge.net/tracker/?group_id=144497&atid=758942* and follow the instructions
- manual: "The Horton Machine", available at *http://www.ing.unitn.it/dica/eng/Quaderni/index.php*

2 An Introduction to GIS

In this manual not much space will be given to the description of Geographic Information Systems and their development over the last number of years. It is assumed that the user that approaches *JGrass* has a smattering of GIS. For this reason only the main terms that will be used in this manual are proposed here.

The introduction to Geographic Information Systems that follows makes reference to the article on the topic at the Italian Wikipedia web site[1], the free encyclopedia. More information is available at the English version of the same encyclopedia[2].

2.1 Introduction

Geographic Information Systems (GIS) are computer systems which are used for representing, displaying, and analyzing the characteristics and elements present on the Earth's surface, as well as the events that take place there.

Any object on the the Earth's surface can be georeferenced, that is to say, its position can be precisely identified on the terrestrial globe. This is the fundamental concept that allows every database to be linked to a GIS. In this manual the term *database* means a set of information about an object and its relation with other "objects", the term *georeferenced* refers to the position of an object in a space defined by a coordinate system.

Before carrying on with the study of GIS, it is important to state that they do not offer pre-packaged solutions for every sort of spatial problem. Everything depends on the skill of the user, it is in fact the skill of the user that puts together different operations in a particular order or according to some particular scheme in order to obtain the best results.

2.1.1 Definitions

A Geographic Information System (GIS) is a computer system capable of acquiring, integrating, storing, analyzing, displaying, and mapping information derived from georeferenced data.

[1] http://it.wikipedia.org/wiki/Sistema_informativo_territoriale
[2] http://en.wikipedia.org/wiki/GIS

2.1.2 GIS and LIS

According to Burrough (1986) a GIS is made up of a series of software tools for the acquisition, storage, extraction, transformation, and display of spatial data from the real world.

It is a computer system able to produce , manage, and analyse spatial data by linking to each geographic element one or more alphanumeric descriptions.

GIS can be seen as a type of DBMS (Database Management System) capable of managing geographical data.

Although the terms GIS and LIS (Land Information System) are often used indiscriminately, many authors agree that the two definitions do not coincide. The term LIS in fact, being referred to land, is restrictive with respect of the term GIS. Mogorovich (1988) defined LIS as: "*The (often informal) body of people, instruments, and procedures that allow for the acquisition and distribution of data within the organization environment and that makes them available in the moment in which they are needed to whoever needs them in order to carry out any activity*".

2.2 Data Model

In order to represent data in an information technology system a flexible and representative model must be formalized which adapts well to real phenomena. In a GIS there are three types of information:

- Geometric: referring to the cartographic representation of the objects represented, such as shape (point, line or polygon), dimension, and geographic position;
- Topological: referring to the reciprocal relationship between the objects (connection, adjacency, inclusion, etc.);
- Informative: referring to the data itself (numerical, text,etc.)

The GIS manages all this information by means of a relational database.

The aspect which characterizes GIS is the geometric one: it stores the position of an object by means of a real projection system which defines the geographic position of the object. A GIS can manage data coming from different projection and reference systems simultaneously(e.g. UTM or Gauss Boaga).

As opposed to cartography on paper, the scale in a GIS is a quality parameter of the data and not of the visualization. The scale value indicates the number of digits of the georeferencing coordinates to be considered significant.

2.3 Types of Geographic Data

The real world can be represented in a Geographic Information System by two main types of data: vector data and raster data.

Vector data are made up of simple elements such as points, lines, and polygons which are encoded and stored on the basis of their coordinates. In a GIS a point is identified by its real coordinates (x1, y1); a line or a polygon by the position of its nodes (x1, y1; x2, y2; ...). An informative database record is associated to each element which contains all the attributes of the represented object.

Raster data represent the real world by means of a matrix of cells, these are usually square or rectangular and are known as pixels. Each pixel contains information about what it represents on the land surface. The pixel size, usually expressed in the same units as the map (metres, kilometres, etc.) is closely related to the precision of the data.

Vector data and raster data are suitable for different uses. Vector maps is particularly suited to representing data that vary in a discrete way (e.g. the location of rubbish bins in a city, street maps, land use maps). On the other hand, raster maps are more suited to representing data that vary continuously (e.g. digital terrain models or a maps of the steepness of slopes).

2.4 Functionality

A GIS allows different data to be put in relation to each other, on the basis of a common geographic reference system, so as to create new information from existing data. A GIS can interact with the user in many ways and a set of tools make personalization and adaptation to specific problems easy.

Normally a GIS offers spatial analysis features, that is to say, the transformation and processing of the geographical elements of the attributes. Some examples of these processes are:

- Topological overlaying: where the superimposition of the elements of two different thematic maps is used to create a new thematic map (e.g. the superimposition of a natural park boundary over a map of the administrative regions so as to determine the jurisdiction of each municipality, or the the percentage of protected area in each municipality);

- Spatial querying, that is the interrogation of data according to spatial criteria (vicinity, inclusion, superimposition, etc.);

- Buffering: defines a polygon around a point, line, or polygon at a fixed or variable distance

as a function of the attributes of the element;

- Segmentation: algorithms usually applied to linear elements in order to determine a point at a specific distance from the beginning of the element;
- Network analysis: algorithm to determine the shortest route between two points of a linear network (e.g. road network).

2.5 Glossary

The definitions in this section make reference to the on-line GIS term glossary, the complete and updated version can be found at: *http://www.glossari.it/glossariogis.html*

A

Attribute - Descriptive information associated to a geographic object which characterizes it. Generally it is used to indicate the non-graphic characteristics of an element or those graphic characteristics which cannot be represented at the scale of acquisition. For a linear element (arc) that represents a road the attributes could be: the type of paving, the average number of vehicles that transit, the width, etc.

Attributes Table - Tables are an integral part of the informative layer. Each table refers to a homogeneous group of geographic elements of the map (roads, rivers, contour lines, etc.) and is made up of a variable number of rows and columns. Each row (record) contains the description of a single geographic element. Each column (field or attribute) stores a specific type of information. The characteristics of the geographic elements are usually expressed in alphanumeric code before being input to the table.

B

Base cartography - It is the set of maps of a certain area, made with topographical or photogrammetric means, that show the planimetry (watercourses, roads, railways, bridges, buildings, etc.) and altimetry, by means of benchmarks or contour lines, of the area.

C

Coordinate System - [see "Reference System"]

D

Database - A database is a set of different types of information. These data are organized according to precise criteria that allow quick consultation.

Digital cartography - In the digital version of a traditional map the position and the description of an element are memorized in a series of files on the computer. To identify the position of an element an x-y or x-y-z cartesian coordinate system is used: each point is represented by a single pair of x-y coordinates; each line is stored as an ordered series of x-y coordinates; each area is stored as an ordered series of x-y coordinates that define the perimeter of the closed figure. By means of the x-y coordinates it is possible to represent points, lines and polygons as lists of coordinates rather than as drawings. In most cases, the storage of elements is done with coordinates from the real world: these coordinates represent a real location on the surface of the Earth, using one of the many coordinate systems. In digital maps the relationship between geographic elements is expressed by means of topology. The characteristics of the geographic elements are stored on the computer in a file, usually called a table, as a series of numbers and letters.

F

Field - In a table of attributes it indicates the part of a record dedicated to a single characteristic of the single elements of the table.

G

Gauss-Boaga - It is a coordinate system based on an inverse cylindrical projection, it is typically used for the cartographic cover of Italy (IGMI Maps, Land Registry Maps, Regional Technical Maps, etc.).

Geographic database - A database dedicated to land and territorial information.

Georeferencing - Computer procedure used to position vectorial and raster data in a real terri-

torial area according to a specific coordinate system by means of points with known coordinates (control points).

GIS (Geographical Information System) - A complex set of hardware, software, and human and intellectual components used to acquire, process, analyse, store, and display in graphic and alphanumeric format data referred to a certain land area.

GPS (Global Positioning System) - A system that, by means of a receiver, software, and a constellation of satellites, allows you to determine the position and altimetry of a point with a precision that goes from a few millimetres to a few metres, depending on the type of equipment and the operative procedures of detection used.

I

Image processing - Set of activities relative to the processing of images in raster format. Many of the techniques used in this discipline require complex mathematical procedures and are aimed at determining the phenomena described by the image, such as pollution, humidity and temperature at the surface, vegetation type, etc.

L

Layer - [see "informative layer"]

Legend - An instrument present on the map which associates the graphic symbols used to individual characteristics and so consents a precise reading of the map.

Longitude - [see "geographic projection"]

M

Map - A map is a graphic representation of geographic elements and other spatial phenomena (lakes, rivers, roads, cities...) that gives two types of information about the area that is represented: position and appearance. The position informs where a certain geographical element is

found on the surface of the Earth and what is its relation with other elements. The appearance illustrates both qualitative and quantitative characteristics of the element represented. Other than the spatial and descriptive information, every map also has some technical characteristics that define it and determine its use. These characteristics include scale, accuracy, projection system, and coordinate system.

Mosaicing - A series of operations, in a GIS environment, by which different digital maps are joined together so as to have territorial continuity.

N

Nominal scale - A parameter that defines the reference scale of a digital map as a function of the corresponding scale of a traditional map made with to the same metric precision requirements.

O

Overlay (topological) - Spatial analysis operation where the informative layers are superimposed and intersected. In this way the information from each layer is joined in a new synthetic layer.

P

Photointerpretation - Technique which allows experts to determine and classify territorial phenomena by reading aerial photographs or other remotely sensed images. Examples include vegetation typology, faultline identification, and the updating of existing cartography.

R

Raster image - Any image can be thought of as being made up of a set of small equal areas (called pixels), arranged in rows and columns so as to form a matrix. The values associated with each cell can express either graphical information (colour, greyscale, etc.)or descriptive information (temperature, slope, etc.).

Rasterization - The operation by which raster images are automatically obtained from vector or printed data (scanner).

Record - A group of contiguous fields of information, which is not necessarily homogeneous and which is accessible in various ways.

Reference System - Every projection has its own reference system, from which the distances are calculated. For example, the Geographic Projection refers to the Greenwich Meridian and Equator intersection. The UTM system refers to predetermined six-degree longitudinal strips which are then subdivided into zones (Italy spreads across zones 32, 33 and 34). The GAUSS-BOAGA system, a totally Italian system, starts from Monte Mario in Rome and uses kilometric coordinates which are conventionally measured starting 1500km left and 2520km right of the reference meridian.

Resolution - It is a local quality parameter of a map. In vectorial maps it corresponds to the dimensions of the smallest detail represented, in rasters it corresponds to the side of the pixels used.

S

Scale - The scale of a geographic map indicates how many times a portion of the Earth's surface was reduced in order to fit onto a sheet of paper. It is usually expressed as a ratio between a distance on the map and the corresponding distance on the ground. The scale used for a map determines the information contained and the dimensions of the area that can be reasonably represented. Maps are defined as having a big scale (1:500, 1:1000, 1:2000), an medium scale (1:5000, 1 :10000), and a small scale (from 1:25000 on).

SIT (Sistema Informativo Territoriale) - Italian acronym corresponding to the English GIS.

Spatial Data - Geometric data characterized by a geographic reference.

SPOT (Système Probatoire d'Observation de la Terre) - A family of multispectral satellites with mobile sighting systems used for remote sensing. The resolution of the Earth is 20 metres for the multispectral satellites and 10 metres for the panchromatic ones. The first launch of this French system dates back to 1986.

T

Thematic Map - Representation of a vectorial or raster map in which points, lines, and surfaces are associated to symbols, hatches, and colours that represent the results of a quality analysis (land use, areas of the land use plan, traffic intensity on a road, etc.).

Topology - A set of rules by which to define in an explicit manner the relationship, connection, and contiguity between spatial elements, and to associate these elements to their relative descriptions (attributes). In a topological data model, for example, it is possible to identify the contiguous areas and to identify the lines that confine each area (boundaries).

V

Vector - It is graphic data archiving system in which the objects are stored on the basis of the cartesian coordinates of the points and lines that make them up.

Vectorization - Operation by which a set of vectors are obtained from a raster image, either automatically or semi-automatically (that is with the assistance of an operator).

3 Installation

JGrass is a multi-platform software, the same version can be installed on any of the main operating systems as long as Java is already installed. Before proceeding with the installation instructions, the basic system requirements necessary for a proper installation of the software are presented. The installation is varies for each operating system but follows the specific standard procedures.

3.1 System Requirements

The system requirements necessary for the running of *JGrass*, version 2.0 are listed in table 3.1

Requirement	**Linux**	**Windows**	**MacOSX**
Processor	1400 MHz	1400 MHz	Power Mac G4 500 MHz
Operating system	any	Windows XP	Tiger
RAM	512 Mb	512 Mb	512 Mb
Free disk space for the software (no data)	90 Mb	90 Mb	90 Mb
Video card	with pbuffer support	with pbuffer support	with pbuffer support
Java Runtime Environment	jre 1.5.05	jre 1.5.05	jre 1.5.05

Table 3.1: System requirements for the running *JGrass*.

3.2 Software Installation

The software is available from the HydroloGIS web site[1] or on autorun CD-ROM. If you have an autorun CD-ROM and the autorun option is active skip to Section 3.2.2 of this manual, otherwise you go to Section 3.2.1.

3.2.1 Jar File Installation

If you downloaded the *JGrass* installer from our web site you will have an archive file with a .jar extension (its default name is jgrass_installer.jar).

[1] www.hydrologis.com

Windows

On some versions of Windows, if you download a ".jar" file this is automatically renamed with a ".zip" extension. If this happens it will not be possible to carry out the installation, first the file must be renamed again with its correct ".jar" extension and you will then be able to proceed with the installation. This is only a little of the enterprising spirit typical of Windows...

The file is an executable java archive and a double-click should launch the installation. However, in some versions of Windows the connection to Java for the the execution is not recognised. In these cases it is necessary to launch the installation from the command line[2] by typing:

```
java -jar jgrass_installer.jar
```

In many cases, however, it is sufficient to set the file as a Java executable application and then double-click on the file icon, the installation will be launched.

Linux

In the Linux environment almost no browser supports the execution of jar files. Therefore, unless you are one of the lucky few, open a console, change the directory to the one containing the installation file, and type:

```
java -jar jgrass_installer.jar
```

MacOSX

The MacOSX environment supports the execution of jar files, it is sufficient to launch the installation by double-clicking on the file icon.

For a stable version of *JGrass* an installation package is always prepared specifically for this operating system. The package is a "dmg" package containing the application. It is sufficient to click on the downloaded package to launch the installation process, just like any other MacOSX application.

3.2.2 The Installation Process

If the installation process has launched correctly the screen display of Figure 3.1 should appear.

[2]The command line in Windows can be accessed by the keyboard shortcut "windows key + r", this launches the "run" window, type "cmd" and hit enter. Once you are at the command line you have to change directory to the one containing the downloaded *JGrass* installation file. An alternative for the less expert user is click on the *start* menu - *accessories* - *MsDOS prompt*

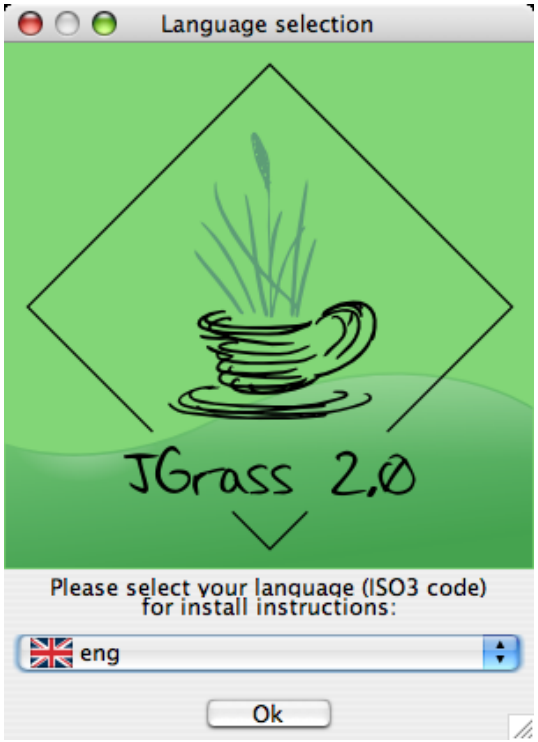

Figure 3.1: The first panel of the installation process.

You will be immediately given the possibility to choose the language used during the installation procedure. It can be chosen from the dropdown menu in the lower part of the panel[3].

The language choice is followed by some informative screen panels. To proceed you only have to click "Next". After the informative panels the user is asked to accept the license chosen for *JGrass*, that is to say a General Public License (GPL)[4].

There follows a screen panel which informs about the changes made with respect to the previous versions of *JGrass*. It also lists the new features of the current version as well as the known bugs. The following screen panel lists some information for the installation of the software in particular situations and with configurations. Generally it is not necessary to follow the guidelines proposed.

The following screen panel (Figure: 3.4) is the most important of the entire installation procedure. It is divided into three parts:

1. Java environment installed: an automatic recognition method for the path and version of the installed java runtime environment has been implemented. If it should fail and the text box should not show the absolute path to the java environment, it will be necessary to insert it

[3]N.B. the logo may be different from the one shown here, depending on which version of the software you are installing.

[4]www.gnu.org

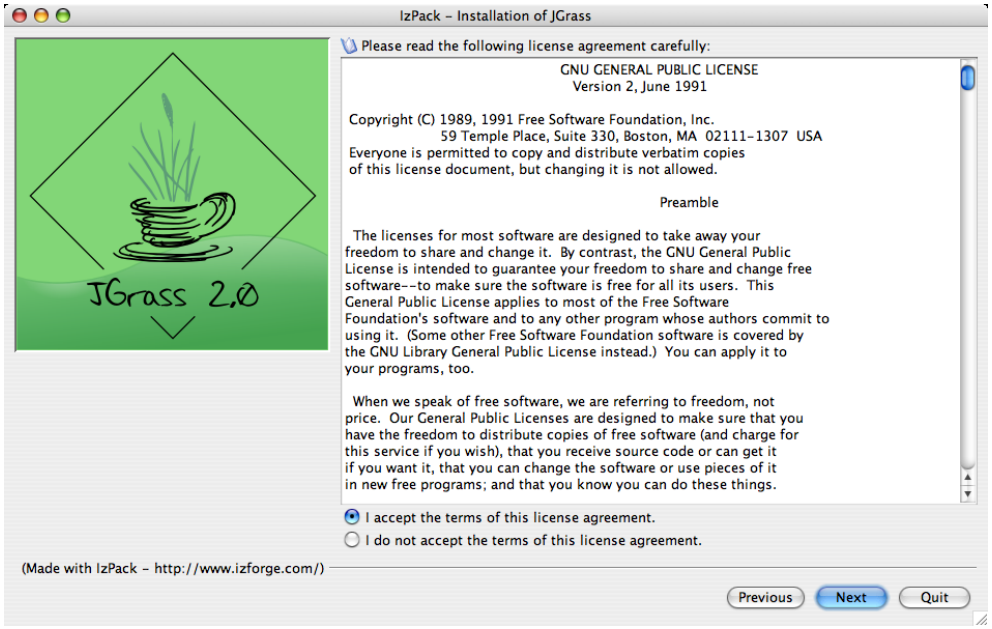
IzPack - Installation of JGrass
Please read the following license agreement carefully:
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place, Suite 330, Boston, MA 02111-1307 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change free
software--to make sure the software is free for all its users. This
General Public License applies to most of the Free Software
Foundation's software and to any other program whose authors commit to
using it. (Some other Free Software Foundation software is covered by
the GNU Library General Public License instead.) You can apply it to
your programs, too.
When we speak of free software, we are referring to freedom, not
price. Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
this service if you wish), that you receive source code or can get it
if you want it, that you can change the software or use pieces of it
in new free programs; and that you know you can do these things.
I accept the terms of this license agreement.
I do not accept the terms of this license agreement.
(Made with IzPack - http://www.izforge.com/)
Previous
Next
Quit
JGrass 2.0

Figure 3.2: Screen shot of the software's license.

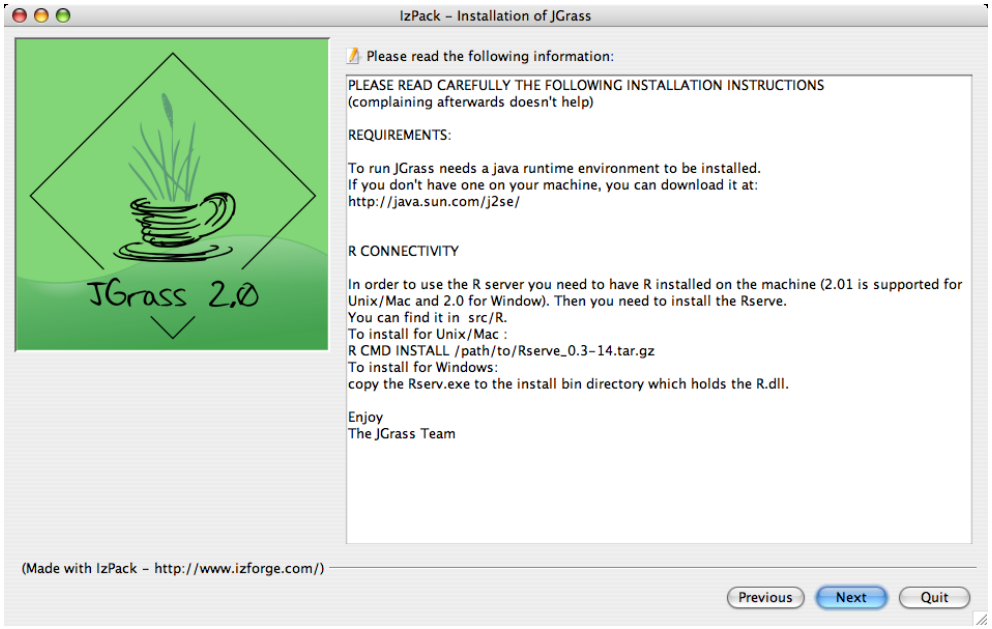
IzPack - Installation of JGrass
Please read the following information:
PLEASE READ CAREFULLY THE FOLLOWING INSTALLATION INSTRUCTIONS
(complaining afterwards doesn't help)
REQUIREMENTS:
To run JGrass needs a java runtime environment to be installed.
If you don't have one on your machine, you can download it at:
http://java.sun.com/j2se/
R CONNECTIVITY
In order to use the R server you need to have R installed on the machine (2.01 is supported for
Unix/Mac and 2.0 for Window). Then you need to install the Rserve.
You can find it in src/R.
To install for Unix/Mac :
R CMD INSTALL /path/to/Rserve_0.3-14.tar.gz
To install for Windows:
copy the Rserv.exe to the install bin directory which holds the R.dll.
Enjoy
The JGrass Team
(Made with IzPack - http://www.izforge.com/)
Previous
Next
Quit
JGrass 2.0

Figure 3.3: The screen panel listing particular installation configurations.

manually[5].

2. Language in which the program will be installed by inserting the corresponding abbreviation.

[5]The path to insert is the one to the "bin" directory which contains the java executable file. In Linux and MacOSX the executable is called "java", while in Windows it is java.exe

3. Minimum and maximum memory parameters of the memory range for the execution of *JGrass*. These two parameters are important as they define the memory resources that will be dedicated to *JGrass*. Setting these parameters well during the installation, taking into account the available RAM, will help avoid unpleasant surprises during the execution, of the type *"Out of memory error"*.

4. Interconnections between *JGrass* and some external programs. If the listed programs are not installed on your system, just leave the contents of the text boxes unaltered (by default there is a point). More details about this will be given in Paragraph 3.3: External Programs.

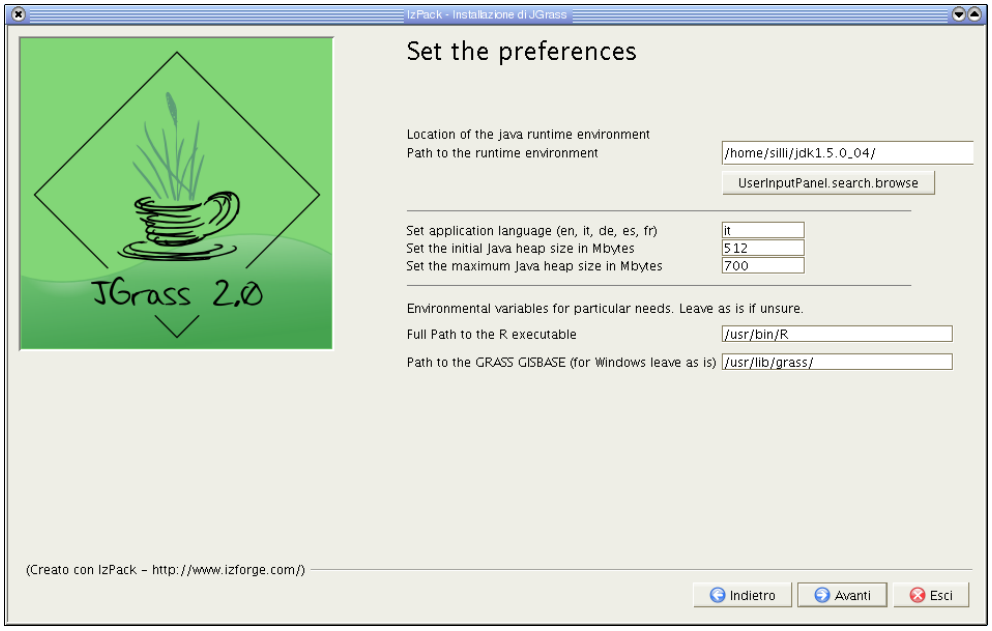

Figure 3.4: The screen panel with the installation options.

In the following panel the chosen parameters are summarized so allowing you one last check before the actual installation. At this point it is still possible to turn back and modify the setup.

In the following step you must select the packages to install (Figure: 3.5):

- binary - the base package for the installation of *JGrass*
- windowsbinaries - package of precompiled GRASS commands for Windows
- src - the *JGrass* source code
- testdataset - a data set to be used for an initial launch of *JGrass*

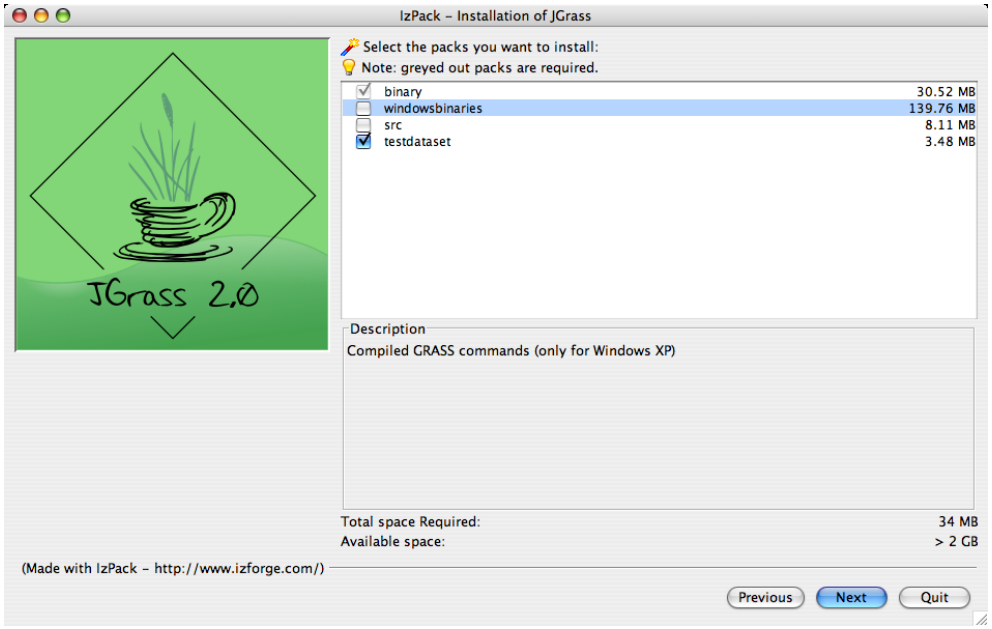

Figure 3.5: The screen panel listing the installable *JGrass* packages.

Once the choices have been made and the "Next" button has been clicked you will be asked to give the *JGrass* installation pathname. If the program is to be installed over a previous version you will be warned that all existing files will be overwritten. If the installation is to a new directory you will be warned that a new directory will be created in the filesystem.

A practical note for the updating of *JGrass*: if the new version of *JGrass* is installed in the same directory as the previous installation the user-defined configurations of the older version will be maintained. In this way you can avoid recreating workspaces. However, this is inconvenient if in the new version new commands and menus have been implemented, in this case the old configurations, being those of the previous version, do not allow you to see the innovations implemented. It will therefore be necessary to cancel the workspace from the list of available workspaces (in this way it is not necessary to cancel the data saved on the hard disk) and recreate it, so loading the new tool bars and commands. To cancel a workspace it is sufficient to right-click on the desired workspace in the initial workspace selection window.

At the end of the installation process, Windows and Linux users have the possibility to include *JGrass* in their start menu (Figure: 3.6) while for MacOSX users an icon is automatically created in the applications bar.

One final panel will inform you of the successful installation and will indicate the path to the uninstall file. To conclude the installation you will only have to click "Done".

The installation is complete and it is now possible to run *JGrass*.

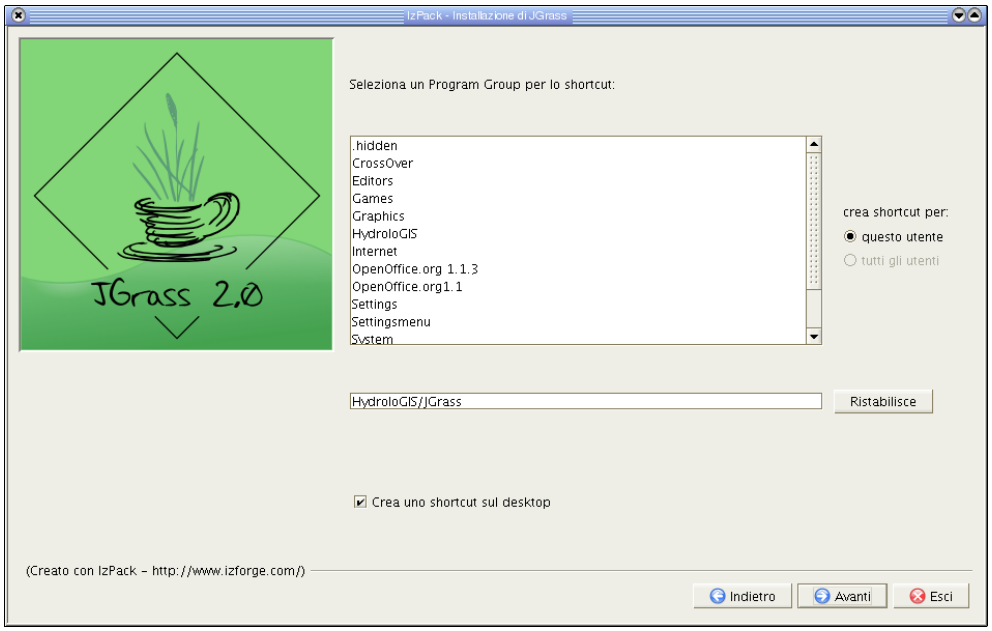

Figure 3.6: Window with the settings for the *JGrass* shortcut.

3.3 External Programs

Two external programs can be linked to *JGrass*: GRASS[6] and R[7]. The interconnections with these programs are managed differently according to the operating system used. In particular, if you use:

- **Windows**:
  - *GRASS*: since version 1.5 of *JGrass* it is possible to install the precompiled GRASS applications for Windows. If you wish to install them (recommended choice) it is necessary to leave as is the selection corresponding to GRASS in the window shown in figure 3.4 and then select the *Binaries for Windows* package for installation in the following window (Figure 3.5).
  - *R*: it is necessary to have the R statistics software already installed on your computer. In the dialog box relative to the installation options shown in figure 3.5 you need to input the complete pathname of the directory containing the R executable file and the connection application for *JGrass*, called *Rserve.exe*, in the space corresponding to R. The text to be inserted is:

    `C:\Documents and Settings\Utente\R\bin\Rserve.exe`

[6] www.grass.itc.it
[7] www.r-project.org

The *Rserve.exe* application can be installed in the following dialog box by selecting the item *R-engine* and later copying it to the directory containing the R executable file.

- **MacOSX and Linux**: both programs must already be installed:
  - *GRASS*: insert the complete installation path for the GRASS commands in the dialog box of figure 3.4 in the space corresponding to GRASS.
  - *R*: insert the path to the R executable file in the dialog box of figure 3.4 at the line corresponding to R. With these operating systems the Rserver package is installed like any other R package by downloading it from the web[8] For this reason only the path to the R executable is needed and not the path to the Rserve package.

3.3.1 The R Packages Used

The programs developed in *JGrass* that rely upon the R connections use specific commands from some R packages. Unfortunately, not all of these packages are included in the basic version of R and therefore they must be installed separately[9] following the specific R procedures, which vary from operating system to operating system[10]. The packages used are:

- akima: it contains the algorithms necessary for the interpolation of spatial data that are distributed irregularly.
- GRASS: the interface between GRASS 5.0 and R
- gridBase: integrates the graphics creation features
- gstat: geostatistical forecasting and simulation models
- sgeostat: An object-orientated Framework work environment for geostatistical modelling
- VR (in some of the newer versions of R this package is included in the basic version...): union of some of the packages of the basic R version such as MASS, class, nnet, spatial

[8] http://stats.math.uni-augsburg.de/Rserve/.

[9] These packages can be downloaded from the web site http://cran.r-project.org/ in the section *Contributed extension packages*

[10] www.r-project.org look in the Manuals section for the R Installation and Administration Manual. The installation of the various packages is explained in chapter 6 for all three operating systems

3.4 Structure of the Installation Directories

Once *JGrass* has been installed the directory JGrass will be created in a position selected by the user. This directory will contain all the files and executables necessary for a proper functioning of the program.

The directories that are created are:

- etc: it contains all the graphic interface configuration files and *JGrass* user profiles
- jars: it contains the Java libraries of the packages that make up *JGrass*
- lib: it contains the native system libraries
- spearfish: it contains the test data package installed during the installation of *JGrass*
- uninstaller: it contains the executable files for uninstalling the application
- grasscommands: it contains the executable files for the native GRASS commands, necessary under the Windows operating system
- src: it contains the source code

Given that the modification of XML configuration files is recommended only to advanced users, the most interesting things to improve and customize the use of *JGrass* are found in the *etc* directory. Attention must be brought to its subdirectories:

- prop: it contains the command and graphic interface configuration files in different languages (e.g. it is possible to modify the number of cells displayed when a map is corrected with the command r.correct by changing the number contained in the file *r_correct.properties*.)
- scripts: it contains all the scripts that are directly executable in *JGrass*
- workspace: contains all the workspace configurations and those of the single user (it must be noted that it is not yet possible to have various users and differentiate access to data according to the user). The most important files in this section are *config.xml* and *workspace.wks*. These files contain the program settings (font size, position of the tool bars, position and size of the scale bar and the legend...) and the settings of workspaces which will be created.

Having said all this, it is advisable that you keep the JGrass directory only as the application "container" while you keep all data you are working with, the so called *workspaces*, in a separate directory on your hard disk, which you can call *workspace*, or *jgrassdata*, *jgrassdb* or simply *data*.

In this way a new version of *JGrass* can be simply installed over the previous version and the data and configurations of the previous version remain unvaried. Only the example data are installed in the main *JGrass* directory because they are installed anew with each new installation and therefore they can be overwritten without any problems.

3.5 Data Structure

The structure with which data are written in order to be processed with *JGrass* is based on the GRASS structure and it has been studied in such a way that you do not have to deal with it directly very often. By all means it is important to know the data structure because it helps during the data import and export procedures.

The data that are used in *JGrass* must be saved in a directory called DATABASE that the user makes on his filesystem independently of *JGrass*. In the DATABASE directory data can be organized according to project or area and saved in subdirectories , called *locations*.

A *location* is defined by its coordinate system, cartographic projection, and the geographic limits within which the work region is found. All subdirectories and files that define a *location* are created automatically when you choose to create a new *location* in *JGrass* (see Paragraph 6.1).

Each *location* contains one or more *mapsets*, which are the directories in which the data are effectively saved. The original reason why the concept of *mapsets* was introduced was to facilitate the multiuser approach to the data. For every user that accessed the data there would have been a corresponding *mapset*. The approach was then kept, not so much to for the multiuser concept but rather because it is possible to save in different mapsets the data relating to different projects or to calculation subregions. From each mapset it is possible to read the maps found in the other mapsets, however, it is not possible to modify them, every modification and elaboration must be saved to the current mapset.

When a new location is created, the PERMANENT *mapset* is automatically created where the fundamental data of the project can be saved, that is to say the data common to all the *mapsets* of the location. Usually data are not used directly from the PERMANENT mapset, rather they are copied to the working mapset where they are successively processed. The general settings of the location are stored in the PERMANENT mapset, in particular the *default region* definition. The boundaries of the *active data region* are defined in each single mapset along with the working resolution required.

The organization and management of the files contained in the *locations* and *mapsets* is done by *JGrass*. All the operations that require the direct manipulation of the internal data structure

should always be done with *JGrass*, only the advanced user should try to manually change the filesystem files directly, but it remains unadvisable to do so under most circumstances. A layout of database use in *JGrass* is shown in figure 3.7, all the items shown correspond to filesystem directories.

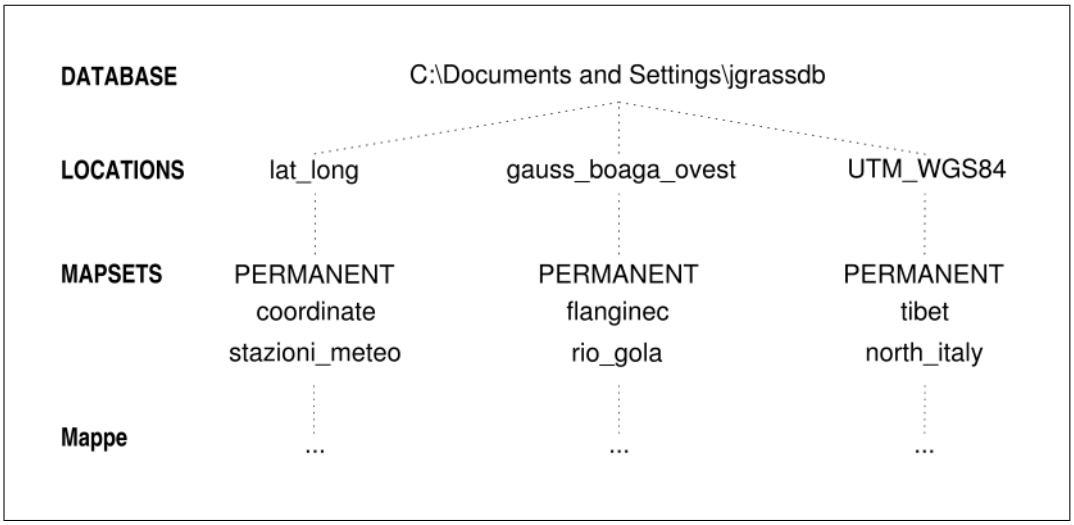

Figure 3.7: Layout of the data structure used by *JGrass*.

In order to understand the *JGrass* data structure it is important to know that all the maps, except the sites, are saved in a series of separate files that contain the data, the categories, the map limits, the resolution, and other information. The directories that are created in a *JGrass location* are the following (here we give a brief description, for further details please consult the GRASS web site[11] or the book, *Open Source GIS: a GRASS Approach*[12]):

1. directories for raster data management, in each one of these directories there are files with particular information corresponding to maps created in *JGrass*:
   - *cats*:contains the files with the *category* specifications created for the maps
   - *cell* and all the directories that have *cell* as part of their name (cell_misc, cellhd, and fcell): they contain all the raster map data
   - *colr*: contains the raster maps colormaps
2. directories for vector data management: these directories are used for the management of vectors in native GRASS format. Usually the *shapefile* standard is used in *JGrass* which

[11] www.grass.itc.it
[12] Neteler Mitasova - Edizioni CAP

does not require a particular data structure in order to be saved to disk.

- *dig*and all the directories that have *dig* as part of their name contain the specifications of the vector maps in native GRASS format
- *dig_ASCII*: contains the vector files in ASCII format which can therefore be consulted directly by the user

3. *JGrass* directories for the management of particular formats of raster maps:
   - *esriasciigrid*: directory with ESRI (ArcView) ASCII files, here they can be read directly in *JGrass* and so you can avoid the import procedure
   - *fluidturtleascii*: directory with Fluidturtle format files (swap format for specific applications developed by CUDAM, in particular the distributed hydrological model GEOtop), here they can be read directly in*JGrass* and so you can avoid the import procedure
   - *grassascii*: directory with GRASS files in ASCII format, here they can be rad directly by *JGrass* and so you can avoid the import procedure
4. *hortonmachine*: directory in which files resulting from hydro-geomorphological analyses carried out with the *Horton* operations within *JGrass* are stored. This directory does not store maps, only the files created by some of the *Horton* processes.
5. *jgrassgroup*: directory with the definitions of the mapgroups (to be used for vertical queries...) created with the *group.create* command . The section relative to groups is still in development and therefore not completely exploitable yet.
6. *site_lists*: directory with the *sites*, that is, the point maps
7. *sql*: directory with data for database connection via sql
8. some files that contain very important information:
   - DEFAULT_WIND: contains the parameters of the *default region*
   - MYNAME: contains the name of the starting location
   - PROJ_INFO: contains information about all the geographic projections of the data
   - PROJ_UNITS: contains the units of measurement used in the projections
   - WIND: contains the parameters of the *active data region*

As you will have noticed, the structure with which data are saved is quite complicated, however, you will rarely have to deal with it directly. If you access the data through *JGrass* there will be no problems as the program knows where to access and save the required data. The only times in which the user accesses the data structure directly is in the case of *JGrass* directories used for particular data formats in order to speed up the import and export processes as described above. To save data in these directories it is sufficient to indicate as destination of the output map the directory corresponding to the required format, or else save in standard GRASS format and then copy the data to the directory corresponding to the required format at a later time. To access the data in these directories it is sufficient to choose the required format initially from the dropdown menu that as default is set to GRASS format.

4 First Start-Up of the Program

4.1 Introduction

To launch the *JGrass* application the following steps must be followed, depending on the operating system in use.

Windows

1. access the Windows Start menu
2. in the "Programs" submenu you will find the "HydroloGIS" submenu
3. within the HydroloGIS submenu click on the *JGrass* icon
4. you can create a shortcut icon directly on your desktop if you wish

Linux (if you have not created an icon on your desktop)

1. access the *JGrass* installation directory
2. launch the executable file jgrass.linux.exe
3. if you wish you can create a link to the executable file wherever you want

MacOSX (if you have not created an icon in the applications bar)

1. access the *JGrass* installation directory
2. launch the executable file jgrass.macosx.exe
3. if you wish you can create a link to the executable file wherever you want

When *JGrass* is launched a check is carried out to see what version of Java is installed. As of *JGrass2.0*, the optimal running of the program is guaranteed only with Java Virtual Machine 1.5 and later versions. This does not mean that the program does not run with previous versions of Java, but there may be some modules and functions that do not run perfectly. If the installed Java version is not the one required by *JGrass* a warning, like the one in figure 4.1, will appear. In the specific case shown, the installed version is *Java 1.5.0_04* while the required version is *Java 1.5.0_05*. The user is given the choice either to proceed to launch the program or to temporarily

shutdown the procedure and install the required Java version, which can be downloaded from *http://java.sun.com/j2se/1.5.0/download.jsp*. If, as in the case shown in figure 4.1, the installed version is a Java 1.5 you can proceed with the launching of *JGrass2.0* as no disfunctions will arise. All you need to do is click on the "OK" button and the program will launch.

Wrong Java Version Error
Java version 1.5.0_05 or higher required. Found java 1.5.0_04 instead.
Please ugrade your Java Runtime Environment installation!
(http://www.java.com)
Continue at your own risk!
OK

Figure 4.1: The *JGrass* warning message for the installed Java version check.

At this point you will have the basic *JGrass* interface on your screen (Figure 4.2).

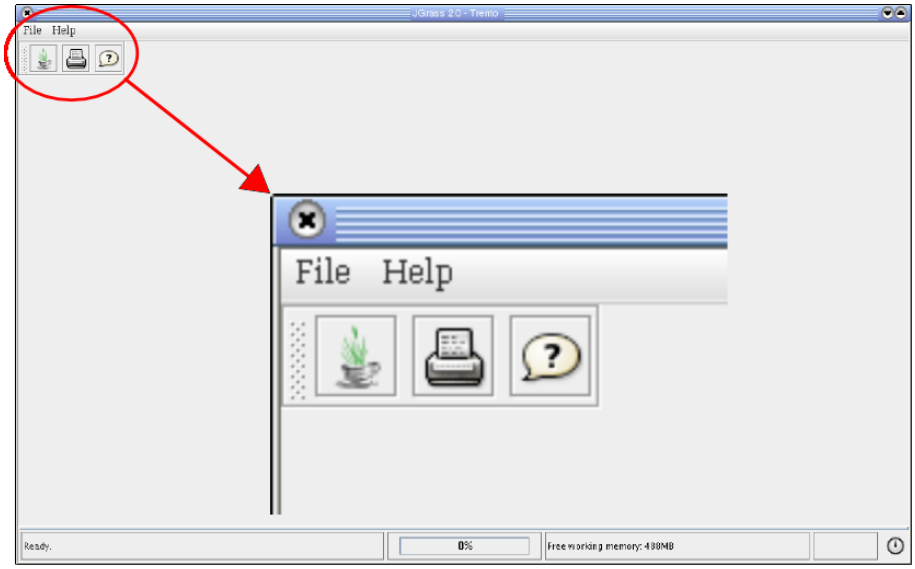

Figure 4.2: The *JGrass* graphic interface at its first start-up with the basic toolbar.

4.2 The Graphic Interface

The *JGrass* interface, shown in figure 4.2, first appears equipped with main menu and toolbar.

4.2.1 The Toolbar

The initial toolbar has three icons:

- load a previously created workspace
- launch the printing dialog box (not usable at this stage)
- launch the manual of commands integrated in *JGrass*

4.2.2 The Main Menu

In the menu bar at the top of the screen is the main menu. It is composed of the *File* and *Help* menus.

In the *File* menu you can find the commands to create and load workspaces, and to define the settings of the program. In the *Preferences*section you can find all the particular and advanced configuration options of the program.

In the *Help* menu you can find the HTML reports that describe the *JGrass* commands, divided according to working categories, and an *About* section where there is some useful information about the program: the names of the developers, the sponsors, and the main characteristics of the program.

If you access the *Help* menu all the available commands can be reviewed. The command classification uses the same scheme as used in GRASS.

- *general*: includes all the GIS general management commands
- *display*: includes all the commands necessary for the visualization of maps and graphics environment management
- *raster*: includes all the commands necessary for analysis of data in raster format
- *vector*: includes all the commands necessary for analyses data in vector format
- *point*: includes all the commands necessary for the management of point data (those data that were called *sites* in GRASS5)
- *group*: commands for the management of group data (in development)
- *hydro*: hydro-geomorphological analysis tools for raster maps

4.2.3 The Workspace

The workspace is a virtual environment in which the resources to be used and the scope of the work to be carried out are defined.

This environment contains the personalized configurations of the main user. Furthermore, there are the menus and the toolbars that the user decides to use. Within the workspace, the geographic limits, the projection used, and the loadable maps are also defined.

As has already been mentioned in the previous chapter, the user personalized configurations are found in the *JGrass* installation directory at the relative path "etc/wokspace/user name". There are two files, *config.xml* which contains the application configurations, and *workspace.wks* which contains the descriptions of the geographical environments, the menus, and the toolbars used.

Opening a Workspace

The *JGrass* installation package provides a predefined workspace. It includes the free data-set called *Spearfish*. By clicking on the first icon on the left of the main *JGrass* window a dialog box opens which lists all the pre-installed and configured workspaces. At the first start-up only one item is listed: "JGRASS test database".

By selecting this item and clicking the "Finish" button the workspace is loaded.

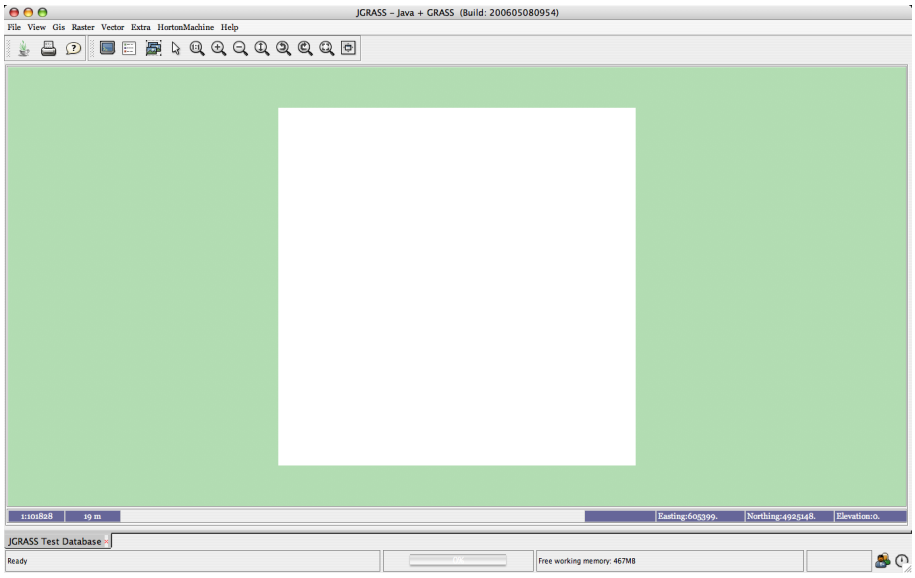

Figure 4.3: Loading the "JGrass Test Database" workspace.

If the operation goes well the main menu will have some more dropdown menus and there will be another toolbar. Furthermore, one part of the screen will be white and the other will be green, as can be seen in figure 4.3.

The green area marks the limits of the computation area for the commands integrated in *JGrass* . The commands that are launched only take into account the area delimited by the green frame, in other words the white area of the screen. The central white area is the so called **active data**

region, while the green area contains the **default region**, in other words the maximum displayable region.

In the lower part of the *JGrass* screen some information which may be useful during processing is displayed, as shown in figure 4.4:

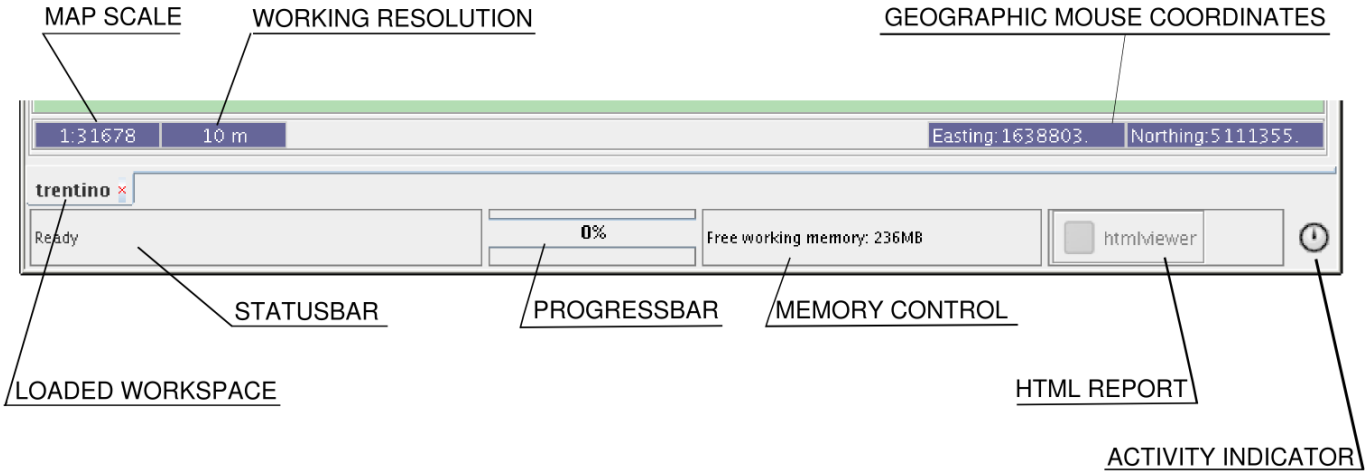

Figure 4.4: Description of the lower part of the screen with indications of the main information shown.

- the map visualization scale
- the current resolution
- the geographical coordinates of the cursor
- the name of the current workspace and the names of the other workspaces that may be open
- the *JGrass* state bar indicating what operation is active at the moment
- a progress bar that indicates the progress of the graphic and numeric processes
- the available RAM check
- names of HTML reports that are open, they can be consulted simply by clicking on this button without having to recall the process that creates them
- a clock-shaped icon that indicates whether the program is in a processing phase or whether it is awaiting new instructions.

4.2.4 The Monitor Toolbar

The display toolbar has twelve icons that launch the following commands:

Figure 4.5: The monitor toolbar.

1. activates/disactivates the console
2. activates/disactivates the legend panel
3. definition of the visible data window
4. sets the mouse pointer to its default configuration
5. sets the zoom to the default region
6. zoom in
7. zoom out
8. zoom by dragging with the mouse:hold the left mouse button and drag the pointer up or down in order to zoom in or out of the map
9. previous zoom
10. next zoom, when you work with map visualizations at different scales it may be useful to alternate the zooms rather than just returning to the previous zoom
11. window zoom, it allows you to define the extent of the visualized area on the screen
12. pan, you can move the map by dragging it with the mouse

We would like to point out that, if your mouse is set in standard mode, it is possible to zoom in or out by scrolling the scroll wheel while you can pan by holding down the scroll wheel and dragging to a new position.

4.2.5 The Menu Bar

The menu bar includes all the operations that can be carried out in *JGrass*. If during installation you chose to load all the tool bars then the main menu will have eight distinct dropdown menus: File, View, Gis, Raster, Vector, Extra, HortonMachine, Help. A description of each now follows.

File

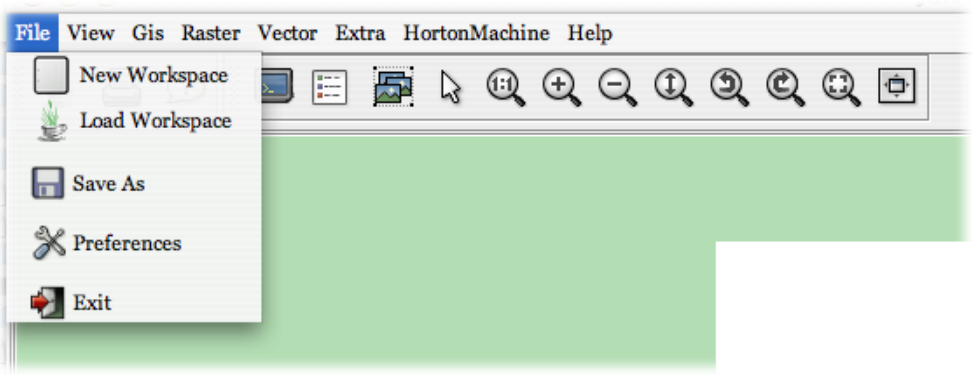

Figure 4.6: The *File* menu.

This menu contains all the tools necessary for the creation of a new workspace, for loading an existing workspace, for saving the settings of the work window, and for changing the settings of the program. See figure 4.6.

For more details on how to customize *JGrass* please refer to Paragraph 4.3.

View

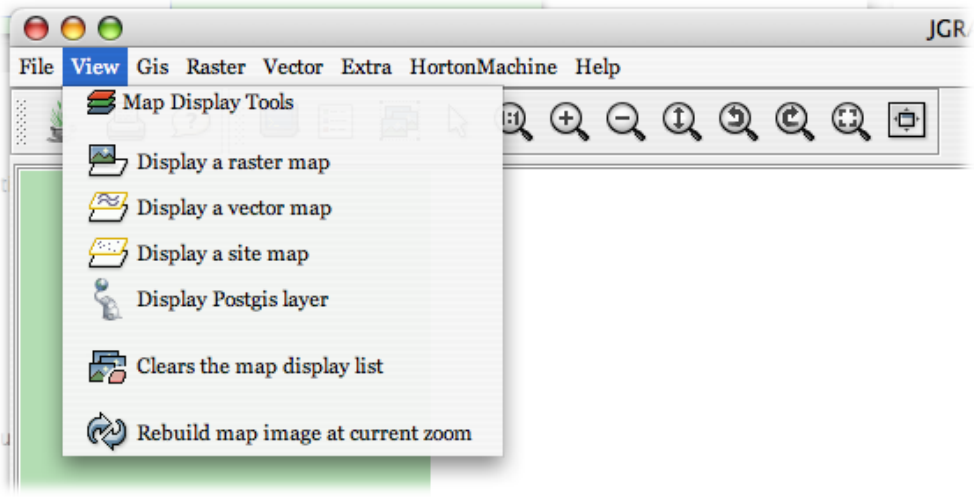

Figure 4.7: The *View* menu options.

The *View* menu contains all the tools for the display of raster and vector maps. By selecting the first item of the dropdown menu, *Map Display Tools* (see figure 4.7), a new tool bar appears (as in figure 4.8).

Figure 4.8: The Map Display Toolbar.

The tools contained in the toolbar are:

- view a raster map by choosing one from the list of workspace maps.
- view vector maps available in shapefile format stored in any directory (they do not need to be moved to the workspace).
- view a point map in native GRASS5 format as a *site* map (these maps are the ones found in the *site_lists* directory introduced in Paragraph 3.5).
- view a vector map from a PostgreSQL remote database.
- clear the screen, eliminating all displayed maps.

- redraw/rebuild all displayed maps. This button is very useful because there are often superimposition problems when various maps are displayed. Should you become aware of such problems you only have to click on the redraw button in order to solve them.

GIS

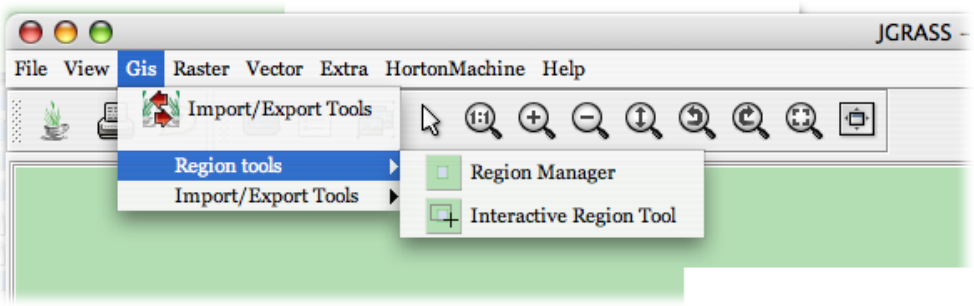

Figure 4.9: The *GIS* menu options.

This section contains all the basic GIS tools, such as tools for the importation and exportation of maps in various formats, including commercial formats, and tools for defining the working region and its properties (see figure 4.9). As has already been mentioned, the term *work/active region* refers to the calculation area, while the term *default region* refers to display limits of the map. The work region must be smaller or equal to the default region, the opposite is not permissable. In *JGrass* region management is left to the user, however, if you try to view a map outside of the default region an error message will appear asking you to modify the limits of your default region so as to include the selected map (see figure 4.10).

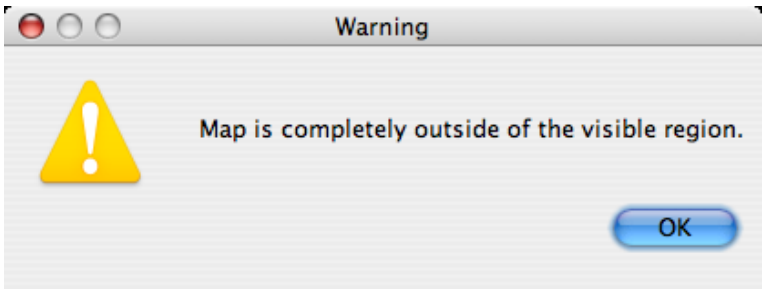

Figure 4.10: Error message when you try to view a map outside of the default region.

In particular, the *JGrass* GIS tools include the following possibilities:

- import external data to *JGrass*. The data formats that can be imported with this algorithm are ArcView ASCII grid format and GRASS ASCII format.

Figure 4.11: The *GIS* toolbar.

- export *JGrass* data to be used by other external applications. The formats that this algorithm supports are GRASS, ArcView, and Surfer ASCII grids. It also gives the possibility of exporting data triplets in x,y,z format.
- import altimetric laser survey data to *JGrass*, and data triplets in x,y,z format in general. The data can be supplied according to a regular grid or as a series of irregularly distributed points on the land surface. In the latter case the data are interpolated, by means of *JGrass*'s *R* connections, so as to obtain regular grid model of the land surface.
- import all Open Source *GDAL*[1] supported data to *JGrass*.
- re-project data from other locations (with different reference systems) in the reference system of the current location. It must be said that re-projecting data normally causes a distortion of the image due precisely to the change of reference system.

[1] www.gdal.org

Raster

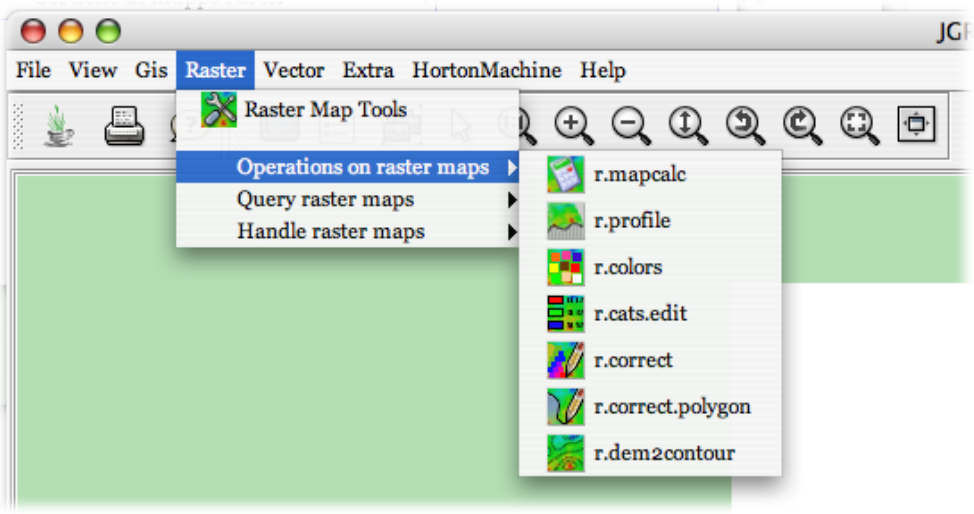

Figure 4.12: The *Raster* menu options.

This section contains all the basic raster management tools. The main functions available (see figure 4.12) are for non-hydro-geomorphological calculations on raster maps, and for modifying map properties (e.g. name or mapset). The options available in the Raster toolbar are:

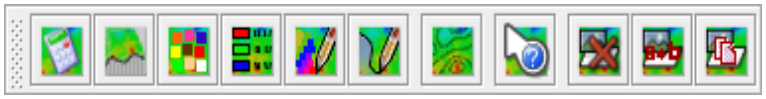

Figure 4.13: The *Raster* toolbar.

- r.mapcalc: to do calculations with maps. The main mathematical operations that can be used have been implemented, such as addition and multiplication/division, but more complex operations, such as the trigonometric ones, have also been included. Furthermore, an algorithm has been implemented to introduce calculation cycles in order to do calculations with various maps.
- r.profile: to plot the profiles of a raster map
- r.colors: to create, modify, or remove the colomap of a raster map.
- r.cats: to create, modify, or remove the data categories of a raster map.
- r.correct: to modify a raster map by editing the values of single pixels.

- r.correctpolygon: to modify a raster map by drawing a polygon within which to change the values of the map.
- r.dem2contour: to create a contour map (vector) from a DEM (raster).
- d.what.rast: to query a raster map.
- r.remove: to remove a raster map.
- r.rename: to rename a raster map.
- r.copy: to copy a raster map to the same mapset, to other mapsets, or to other directories.

As has already been mentioned in previous chapter, an important aspect of the *r.copy* command is that you can copy raster maps to directories other than the standard *JGrass* directories. In this way you can export maps directly with this command without having to resort to specific export commands. In particular, within the *workspace* directory created by *JGrass* there are other directories, as described in Paragraph 3.5, called *esriasciigrid*, *fluidturtleascii*, *grassascii*. Copying a raster into one of these directories means exporting the file into the format specified by the name of destination directory.

Vector

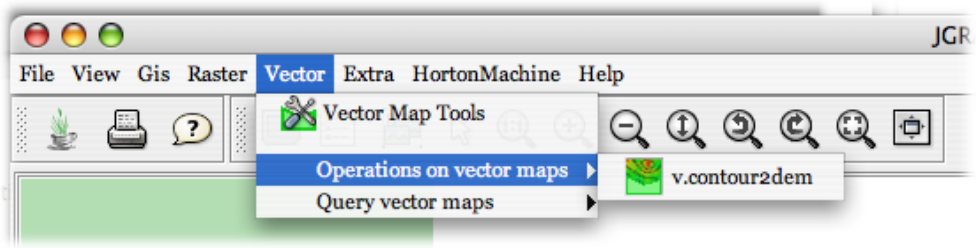

Figure 4.14: The *Vector* menu options.

Originally, *JGrass* only dealt with raster maps, and for a long time it was a distributed GIS dedicated solely to hydro-geomorphological environmental analysis. Future developments include the development of a vector data handling environment which goes from displaying maps to inputting and modifying points, lines, and polygons. For the development of this we have chosen

to follow the Open GIS Consortium standards which propose the *shapefile* format as the saving and swap format for vector maps. This policy is somewhat in contrast with GRASS policy, total compatibility is guaranteed, however, with the vector part of GRASS5,but total compatibility is not yet available with GRASS6.

In this section there are only two commands at the moment, the one that allows you to query the maps and so "see" the associated data table (see figure 4.14), and a real vector analysis tool which allows you to convert a contour-line map into a DEM (raster) by means of interpolation. The command is *v.contour2dem* and the corresponding icon, at the moment only present in the menu, is .

Vector map querying allows you to obtain, by clicking on the desired object, all the information relative to the selected object, be this a point, a line, or a polygon. To view the table with all the vector information it is sufficient to right-click on the name of the displayed map in the *legend* part of the screen and select the *View data* option, as shown in figure 4.15.

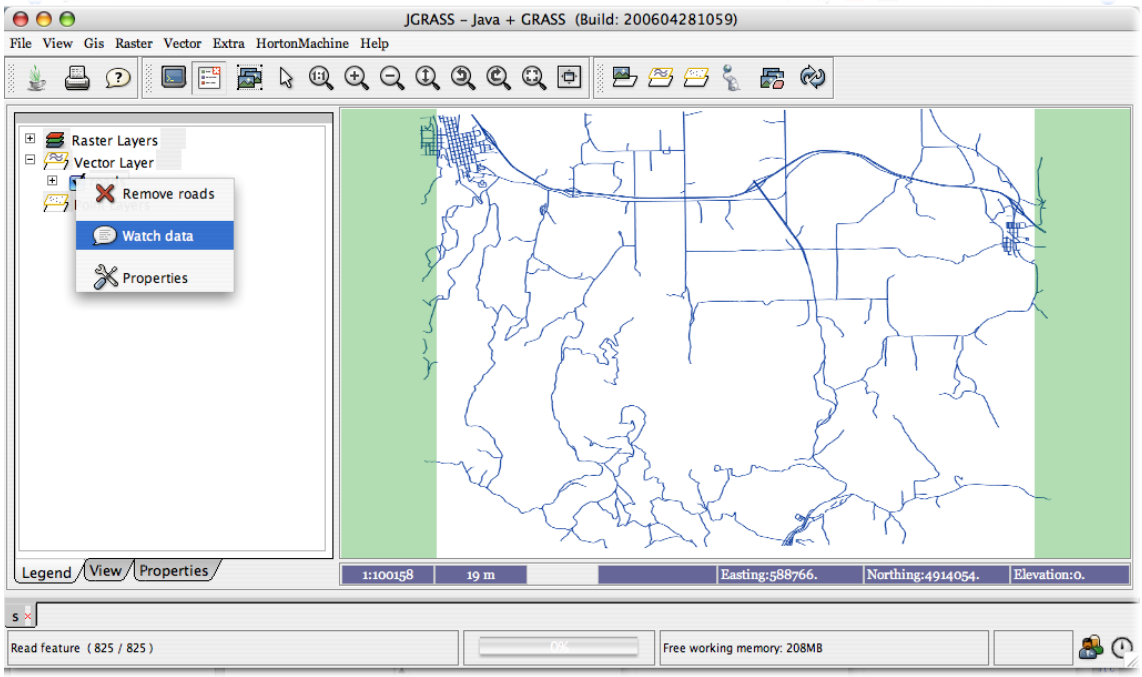

Figure 4.15: Viewing a shapefile in *JGrass* and the possibility of viewing the contained data.

Extra

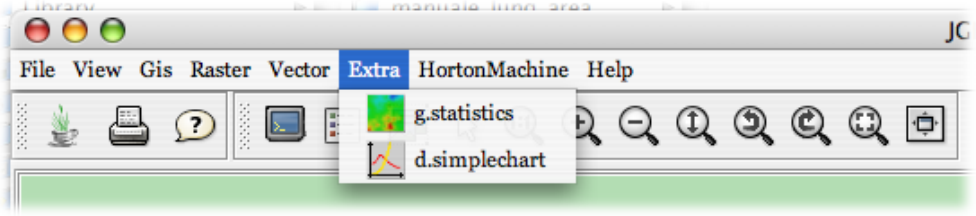

Figure 4.16: The *Extra* menu options.

This section holds the *Extra* tools integrated in *JGrass*. These tools include statistical tools, that can be used through the connection with the Open Source statistical analysis software *R*, and graphic display tools. What has been integrated so far is shown in figure 4.16.

HortonMachine

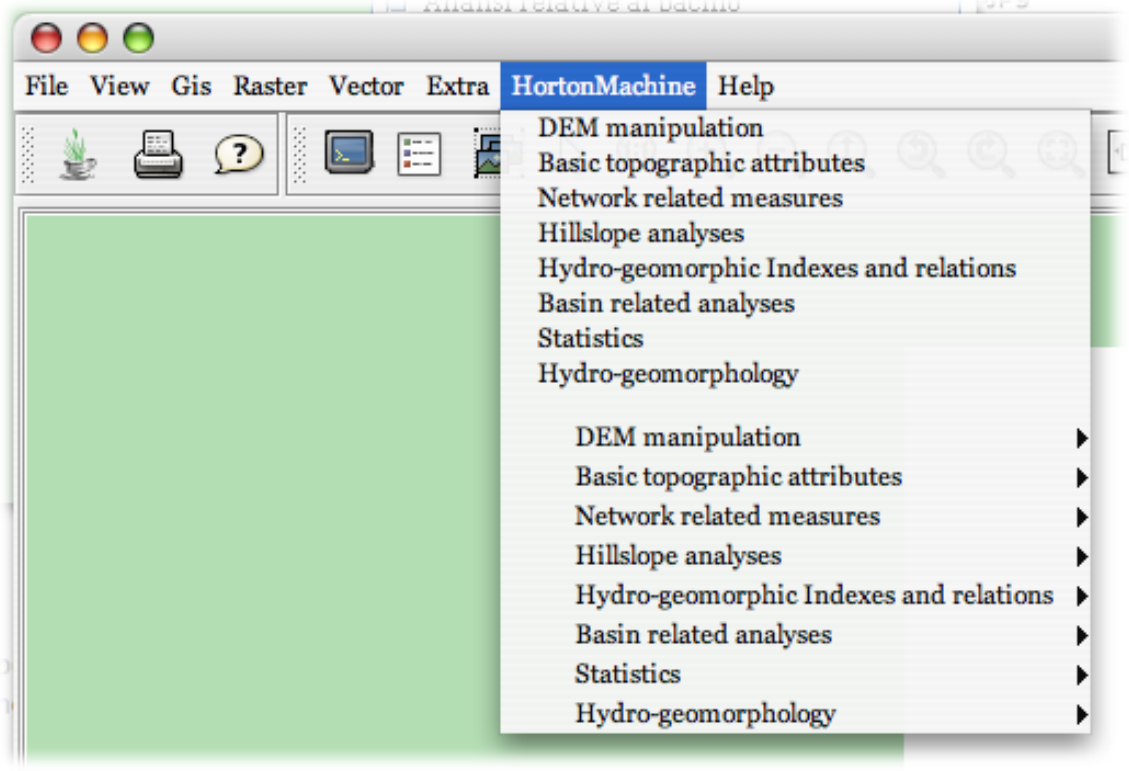

Figure 4.17: The *HortonMachine* menu options.

The hydro-geomorphological processing section contains all the territorial analysis algorithms necessary for delimiting areas that are hydro-geomorphologically at risk.

The options of this menu are not dealt with in detail here, for a complete definition of the

available options you can refer to the HortonMachine Tutorial and "The Horton Manual".

Help

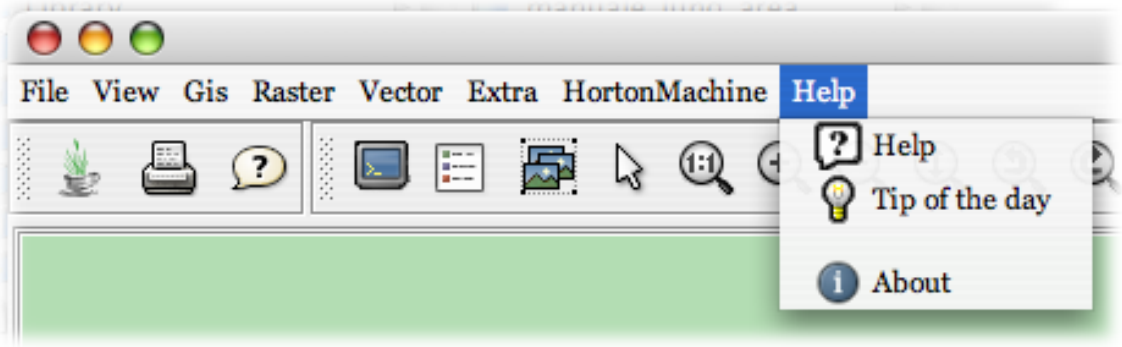

Figure 4.18: The *Help* menu options.

In this section there are the main indications for the commands, and some particular references to the developers, sponsors, and the Open Source *JGrass* Project participants. The contents of the menu, already mentioned in Paragraph 4.2.2, are shown in figure 4.18.

4.3 Customization

The *JGrass* graphics environment and all the configurations for the connection to external programs are configurable by the user by selecting *Preferences* from the *File* menu (see figure 4.6).

The *Preferences* option should be only accessed by advanced users, and by all means the default settings are valid. If you decide to proceed with the customization procedure a window, like the one shown in figure 4.19, will appear with the following tabs:

- *Framework*: graphic interface management
- *GRASS Environment*: GRASS connection configuration
- *Monitor*: display properties management
- *Cluster*: This option is not yet active
- *Console*: *JGrass* console configuration management

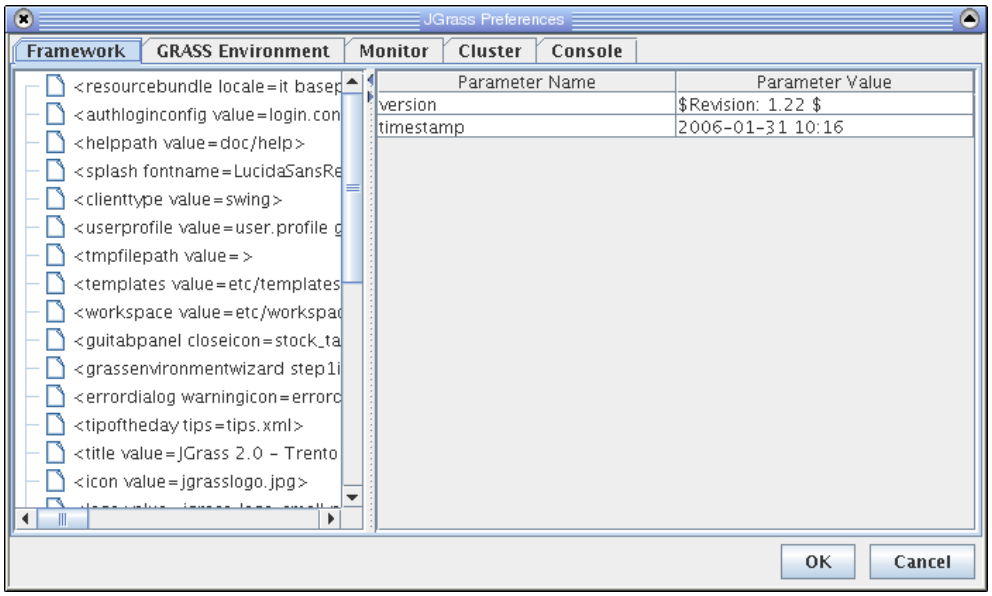

Figure 4.19: The initial window of the *Preferences* configuration panel.

Framework

It is the configuration starting point, here it is possible, for example, to configure the font definitions, size and type, and the display language used. In particular we would like to point out:

- `resourcebundle locale = it`: this indicates that the graphic interface language is Italian, it is possible to choose from the languages available in *JGrass*: English (en), Spanish (es), German (de), and French (fr)(some items in the French version are still in English because the development is still incomplete, if you know French you might like to contribute...).
- `template value = etc/template`: location of the printing templates defined by the user
- `workspace value = etc/workspace`: location of the workspace settings

GRASS Environment

In this section the definitions of the connections to the GRASS commands can be viewed. As has been explained in Paragraph 3.3, a mechanism has been developed by which it is possible to execute native GRASS commands from within *JGrass* on all operating systems. The contents of this window relative to the Linux and MacOSX systems is shown in figure 4.20, while in a Windows environment the window is like the one shown in figure 4.21. As can be noted, the options set during installation can be modified as well as some standard *JGrass* definitions. In particular:

- `gisbase`: reference path to the GRASS6.0 executable commands. Here lies the substantial

difference between operating systems, under Linux or MacOSX, where GRASS must be installed separately, the path to the GRASS installation must be inserted in this box, under Windows, where the command binaries are supplied, it is sufficient to leave the point (.), exactly as explained during the installation phase of the program.

- `scriptpath`: directory where the executable *JGrass* scripts are to be stored
- `libpath`: path to the GRASS libraries
- `binpath`: path to the GRASS command binaries
- `esrigridpath`: directory where files in ESRI ASCII Grid format are to be saved for their direct management with *JGrass*
- `fluidturtleasciipath`: directory where files in FluidTurtle format (a swap format between *JGrass* and GEOtop, a catchment area hydrological balance application used mainly for small mountain catchments) are to be saved for their direct management with *JGrass*
- `grassasciipath`: directory where files in GRASS ASCII format are to be saved for their direct management with *JGrass*
- `shapepath`: directory where files in *shapefile* format are to stored in order to be viewed in *JGrass* by means of the *d.vect* command. This option has been outdated with the introduction of the new *d.shapefile* command which allows you to view this type of file saved in any directory of the filesystem.
- `sqldbpath`: directory where the files containing the query definitions for the database are saved. Even this option is partially outdated by the direct connection to the PostGIS database.
- `jgrassgrouppath`: directory used for the management of groups of raster maps created in *JGrass*. Under development.
- `commands`: name of the file with the definitions of the executable *JGrass* commands.
- `epsgdata`: path to file with the cartographic projection definitions given by the OGP Surveying and Positioning Committee[2].

[2] www.epsg.org

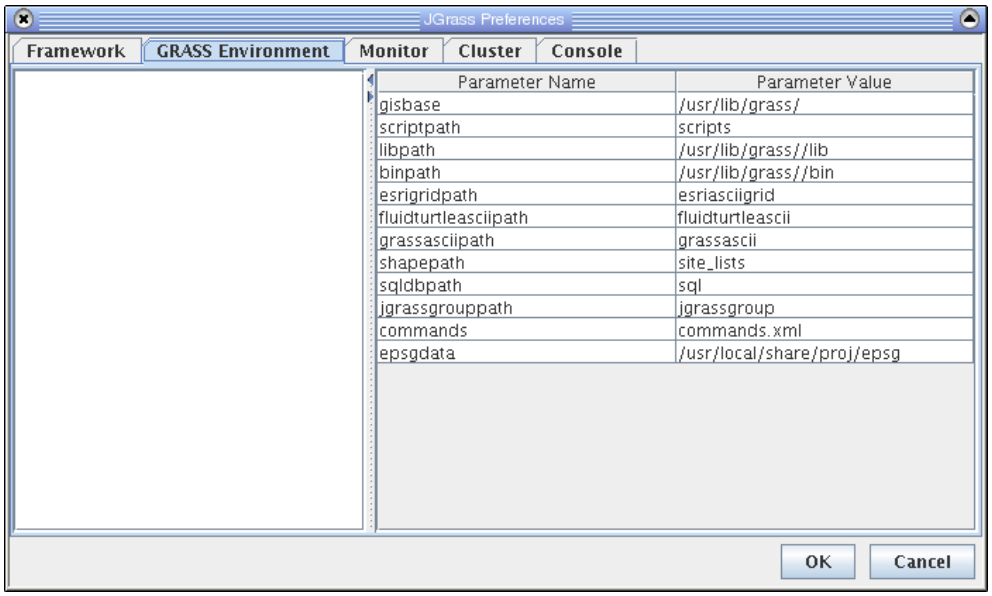

Figure 4.20: The GRASS configuration panel window under the Linux and MacOSX operating systems.

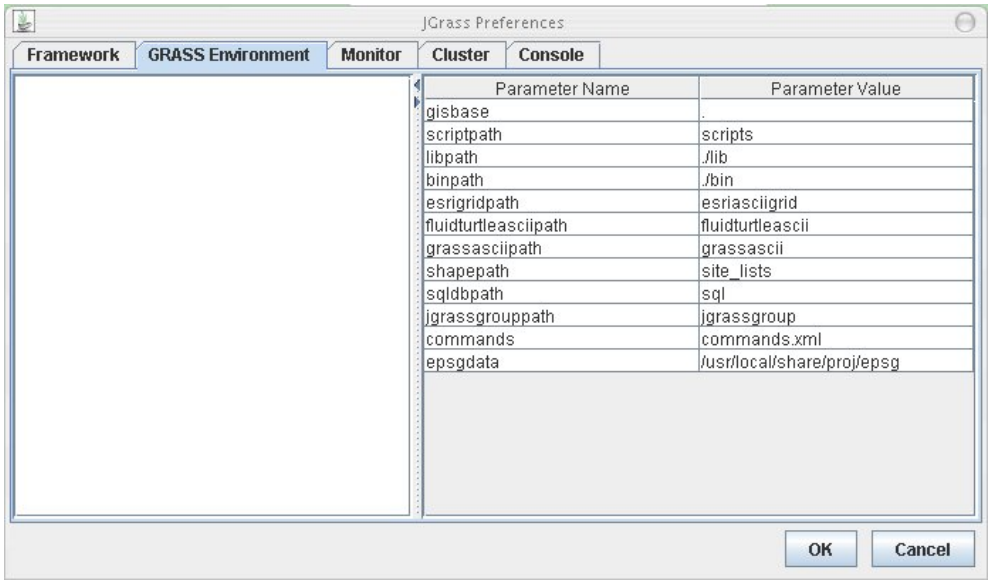

Figure 4.21: The GRASS configuration panel window under the Windows operating system.

Monitor

In this section the display options are defined. We again point out that many of these specifications are only modifiable by those users that are very familiar with both the operating system and the application program. What you will find in this section is shown in figure 4.22.

The most important thing to highlight in this section is the possibility to change the graphics environment "engine". *JGrass* may present display problems for certain combinations of operating system and graphics card when the main graphics engine, that is to say OpenGL, is not optimized.

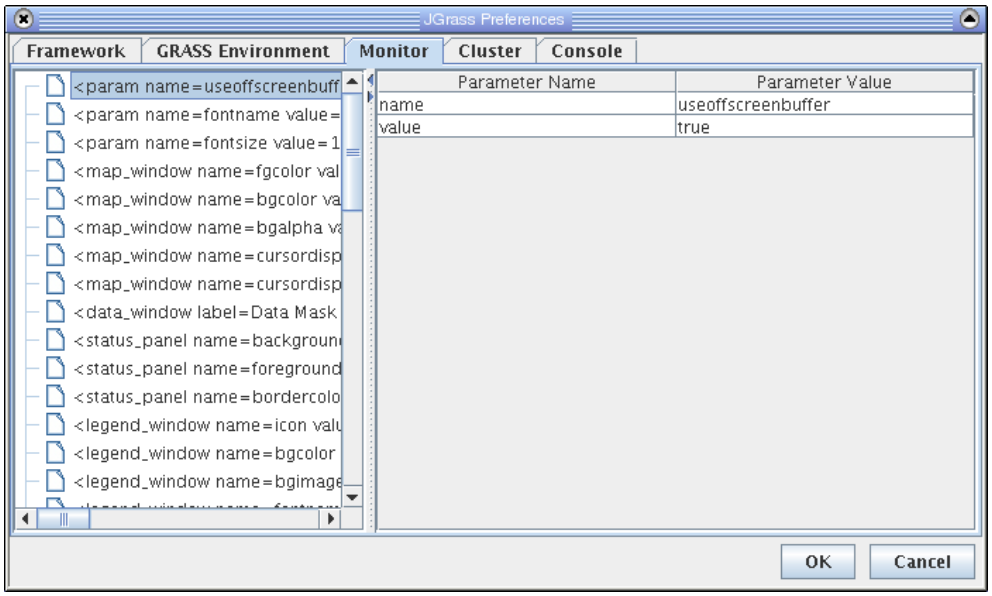

Figure 4.22: The *Monitor* configuration panel window.

If there are display problems, you can pass from the OpenGL based display to the Java2D display (slightly slower). This change can be carried out by simply writing *false* in correspondence of `Value` in the `Useoffscreenbuffer` section, as shown in figure 4.22.

Console

As shown in figure 4.23, in this section you can modify the console configurations, the font, the colour of the text and the background, and the command prompt mode.

4.4 Display Properties

Some map display properties can be defined by simply right-clicking in the main *JGrass* window (the green and white one). A popup menu appears in which it is possible to toggle certain particulars which are especially useful at the printing stage.

These particulars are:

- **Map Scale**: displays the map scale in the main window. The scale is not in the classic form (e.g. 1 : 10000), but rather in graphic form, that is to say, a reference length is given by which to evaluate the dimensions of the map.

- **North Arrow**: displays the North arrow in the main window.

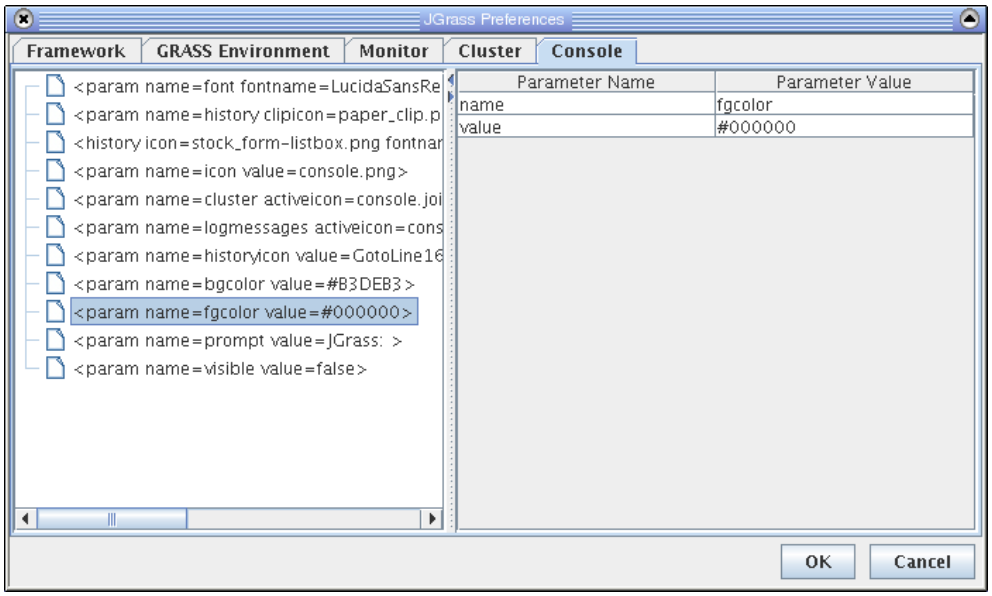

Figure 4.23: The *Console* configuration panel window.

Figure 4.24: The *Map Display Properties* manu.

- **Data Window**: displays the outer limits of the active region by marking in green (default) for what lies outside the region. By deactivating this option the entire window is white even though only a part of it is the active data region.
- **Legend**: allows you to eliminate the legend from the display and to view it only when all the necessary options have been configured
- **Automatic Redraw**: activates the automatic redraw function so that all maps are redrawn every time an operation that changes the display is carried out, for example a change of zoom or a new map being loaded.

The configuration of the the elements selected can not be modified from this menu, for this operation please refer to Paragraph 5.3.1 - *Display Options*.

5 First Steps with JGrass

5.1 The Spearfish Test Data Set

The "free" Spearfish data package, which is installed automatically with *JGrass*, contains raster, vector, and point data. The name derives from the name of the area in South Dakota (USA) to which the data refer.

The data are given in a UTM conic projection, with Clarke66 ellipsoid and NAD27 datum. These projection data, which may seem strange to an Italian user, are of common usage in the USA. Furthermore, the type of projection used is not a fundamental factor while working within a single workspace, as long as it is coherent with the data being used, it does become fundamentally important when working with maps from different workspaces.

5.2 Region Parameters

If you want to view the boundaries of the active data region, the white area of the screen, and the default region, the greatest viewable area, you must select *Region tools* and then *Region Manager* from the *GIS* menu. The dialog box shown in figure 5.1 will appear. The upper part of the window refers to the active data region while the lower part refers to the default region.

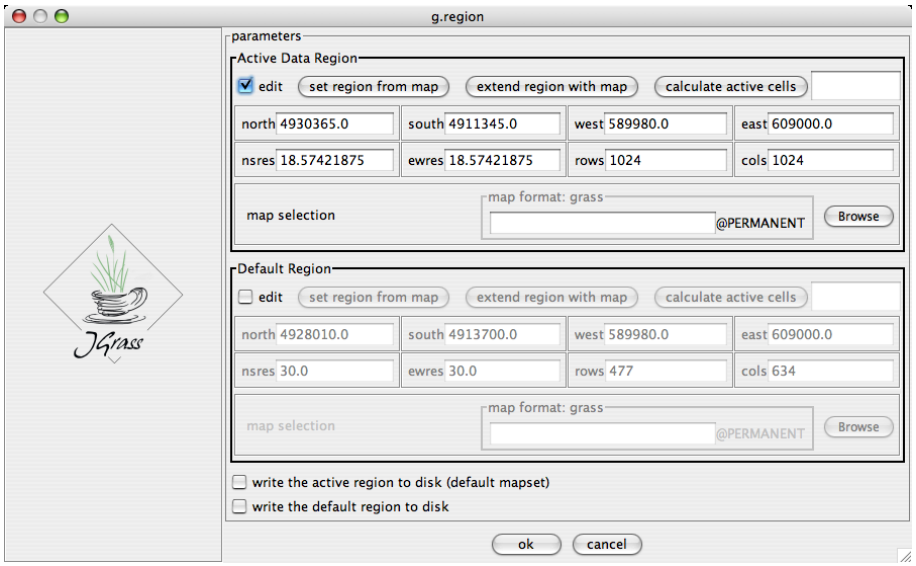

Figure 5.1: The window the current active data region and default region information.

Assuming that initially the default region does not need to be modified by the user, the possibility

of editing it is deactivated, in order to proceed it is sufficient to check the corresponding *edit* check box.

The information contained in the region window are:

- *Set region from map*: to set the active data region to that of a saved map. This option only works if you select a map in the *map selection* box.
- *Extend region with map*: to extend the active data region to include a map that is partially or totally outside the region.
- *Calculate active cells*: to calculate the number of valid values in the active data region of a map . The option only works if you choose a map from the *map selection* menu. The number of active cells is useful in the calculation of the map area within the active data region. The area is obtained by multiplying the number of active cells by the resolution in x and y.
- *Region limits*: coordinates of the region limits. The option only works if you select a map in the *map selection* box.
- *Resolution*: map pixel dimensions in x and y.
- *Number of rows and columns*: number of rows and columns of the map at the displayed resolution.
- *Map selection*: selection box for the map from which to set the region, extend it, or from which to calculate the active cells.

Everything that has been stated for the active data region is also valid for the default region once the *edit* check box has been checked.

The edited values can be saved to disk. The saving, however, is not automatic and the user must check the appropriate check box in the lower part of the window to save both the active data region configuration and the default region configuration. If the changes are not saved the changes will be active during the current work session but they will be cancelled upon the next start-up of *JGrass*.

5.2.1 Interactive region tool

If, from the *GIS - Region tools* menu, you select *Interactive Region Tool* it is possible to change the region boundary parameters with simple graphic operations using the mouse, rather than having to insert the data manually in the dialog boxes of the interface shown in figure 5.1. The toolbar that appears on-screen by selecting this option is shown in figure 5.2.

Figure 5.2: The *Interactive Region Tool* toolbar.

During this stage you can select the boundaries to modify and select the new desired position by clicking on the screen, you can select an area that you want as active region (fifth button from the left), you can move the active window to a new desired position, and you can set the map resolution.

Warning: this toolbar must be closed by clicking on the red **x** button, on the upper right-hand side, once the boundary limits have been set and by all means before carrying out any other operation in *JGrass*. Keeping the toolbar open could cause problems with the following operations because of the superimposition of different types of processes.

5.3 Viewing the Data

The data in Spearfish can be viewed with the aid of the map display toolbar.

5.3.1 Raster Data

By clicking on the icon the *Raster Map Wizard* for the viewing of raster maps is launched. The first panel gives you the possibility to choose the data type you want to display. The dropdown menu lists the different display options (see figure 5.3).

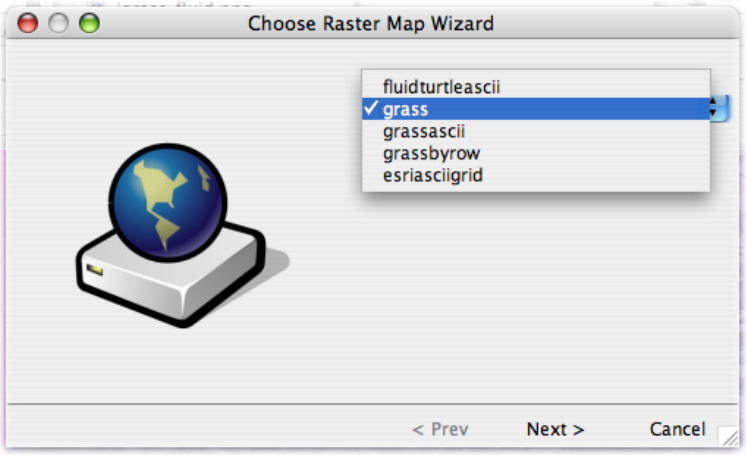

Figure 5.3: The first panel of the raster map wizard.

In this specific case there are maps only in the GRASS directory, they are saved according to the logic used in GRASS. This is the standard mode in which *JGrass* functions, if you wish to load data from one of the other directories (grassascii, esriasciigrid o fluidturtleascii) it is sufficient to choose the corresponding name from the dropdown menu.

Leaving the GRASS option selected and clicking on the *Next* button, a second panel appears from where you can choose the mapset where the required map is saved.

This option is typical of *JGrass* because the program can work in multi-mapset mode within a single workspace. This mode is very convenient during the defining stage of your workspace and during the choice of initial maps to be processed, insomuch that it is possible to access various data sources. However, we advise you to use workspaces with various mapsets only at this stage and to then proceed with workspaces with only one mapset. In this way you can use the native GRASS commands without problems (see Paragraph 6.1) and "save" time during data processing. Otherwise it would be necessary to select the required mapset every single time, both for the loading and saving of maps. To view the list of maps in a mapset, in alphabetical order, just click on the

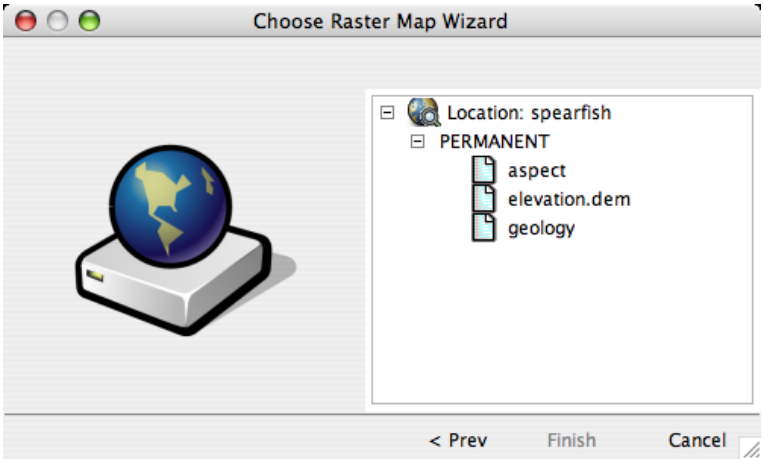

Figure 5.4: The second panel of the raster map wizard, list of available maps.

+ sign in front of the desired mapset. For the Spearfish data set it is sufficient to click on the + sign before the *PERMANENT* mapset, a list of the maps in the mapset will then be displayed, as shown in figure 5.4. At this stage all you have to do is select the map that you wish to view (in the example the selected map is *elevation.dem*) and click *Next*.

In *JGrass* maps are not displayed in successive parts, but rather in one single block, it is therefore necessary that the program read the entire map, load it to memory, and then display it. The progress of the map display operation is shown in the *JGrass Progressbar*, in the lower part of the screen (see figure 4.4). The map chosen for display in the example should appear as in figure 5.5.

If you repeat the display procedure choosing the *aspect* map you will obtain a superimposition of maps as shown in figure 5.6. The aspect map maps the slope exposure, or aspect, and it is often displayed as a base for image processing in *JGrass* because it gives the idea of working in a 3D environment.

In this way all the maps in the loaded mapset can be viewed. In *JGrass* any map that is successively chosen for display will be superimposed over the maps already displayed, each map is assigned a certain transparency factor in order to superimpose as many maps as possible.

Warning: with *JGrass2.0* many of the display problems associated with the video card have been solved. However, if under Windows you should encounter map display problems or should the program shut down during the visualization of a map you will have to reduce the graphics acceleration of the video card.

Warning: in some cases the visualization of new maps can cause an offset of the display, to

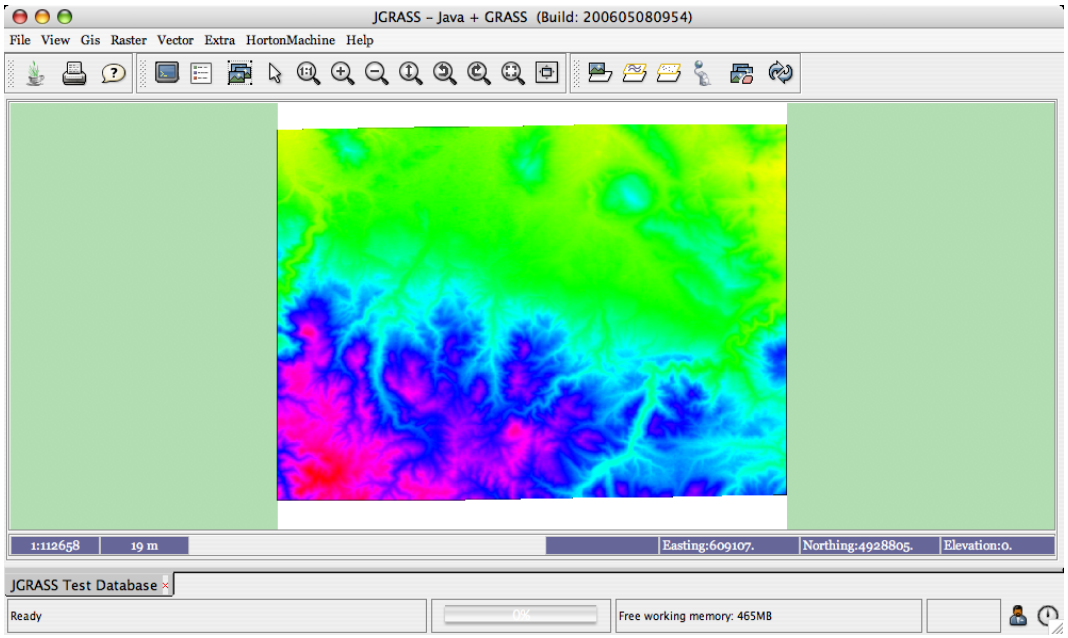

Figure 5.5: *JGrass* view of the elevation map from the Spearfish dataset

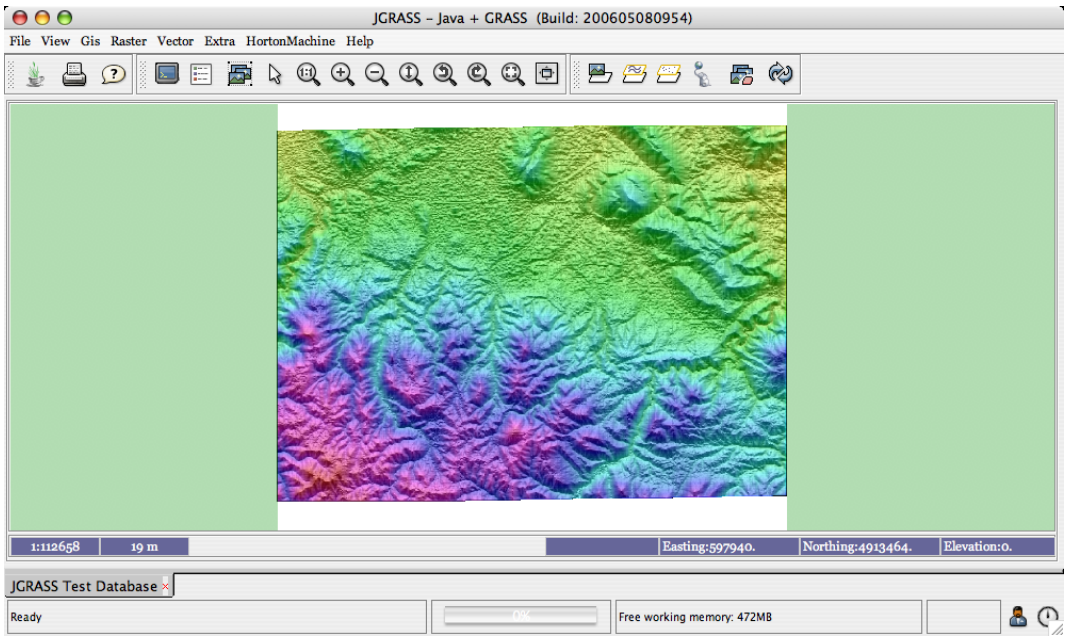

Figure 5.6: *JGrass* view of the elevation and aspect maps from the Spearfish dataset

solve this it is sufficient to reload the visualization by clicking on the button and everything will be redrawn and displayed correctly.This operation is also necessary every time that you use a particularly different zoom or a particularly different view in order to adjust the display to the data that have actually been loaded. It is possible to activate the *automatic redraw* mode, as

explained in Paragraph 4.4, so that these operations are carried out automatically by the program. This choice is convenient if you are dealing with few data, it slows everything down if you have to manage a lot of data or if you have a lot of maps loaded.

Display Options

To control the map display options it is necessary to display the *Legend* window by clicking on the button. In this window you can decide whether to display a loaded map or not by simply checking the check box in correspondence of the map name. You can also remove loaded maps from memory to speed up the visualization process when they are no longer needed. To remove a loaded map it is sufficient to right-click on the name of the map and choose the option *Remove "map name"*. This operation removes data from RAM, it does not remove data from the filesystem. In this way you can avoid overloading the memory of your computer which would slow down the display operations and calculations. The maps that are removed can be displayed again at a later stage, should the need arise, by loading them from the hard disk. By right-clicking on the name of the map you can also define the transparency to assign to it. Figure 5.7 shows an example where the *geology* map has been loaded but is not displayed. The same figure shows the popup menu that appears by right-clicking on the *aspect* map and the window that appears by selecting *Properties* from this menu. The raster's (*alpha*) transparency parameter is defined in this window, a value of 1 indicates that the map is not transparent, the transparency is increased by reducing the value of the *alpha* factor.

Display properties

With the *Legend* window displayed you can define many map display properties that have already been presented in Paragraph 4.4. You can do this by clicking on the *Properties* tab at the bottom of the panel.

In this window the same options as seen in figure 5.7 are proposed again, and for each one of these it is possible to customize the specified default parameters. To do this you will need to view the contents of each option by clicking on the + that precedes the name and then right-clicking on the desired option. From the *Proprieties* menu i t is then possible to define the new parameters. Figure 5.8 shows a display example with:

- Map grid displayed with a NS and EW resolution of 1000m.
- North arrow displayed

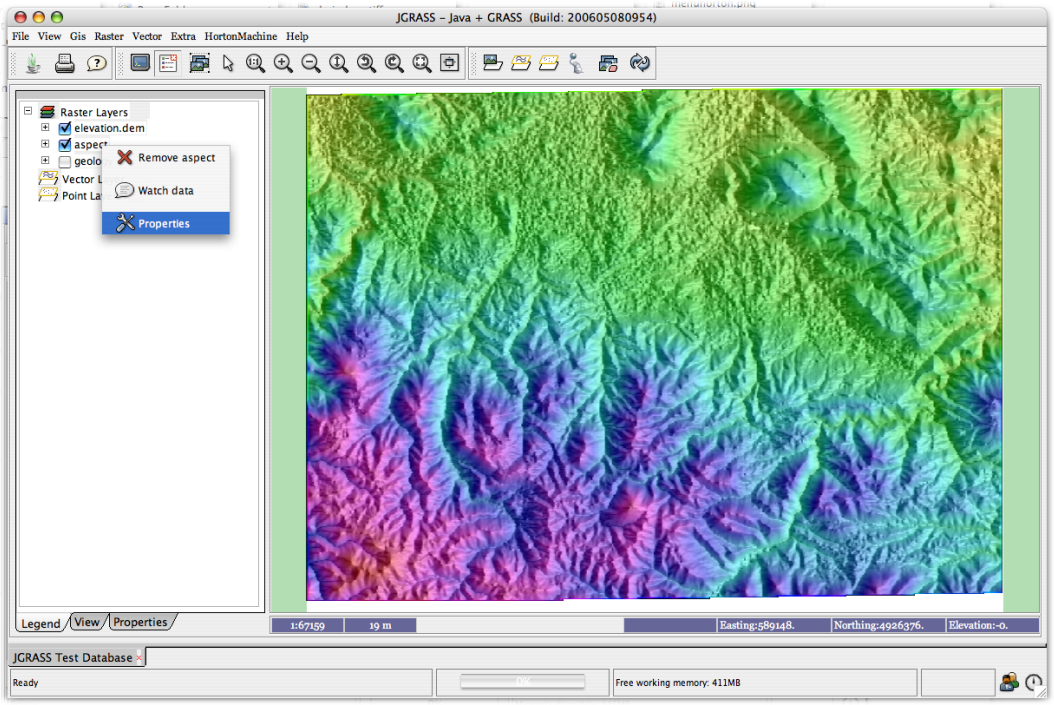

Figure 5.7: *JGrass* display options for loaded raster maps

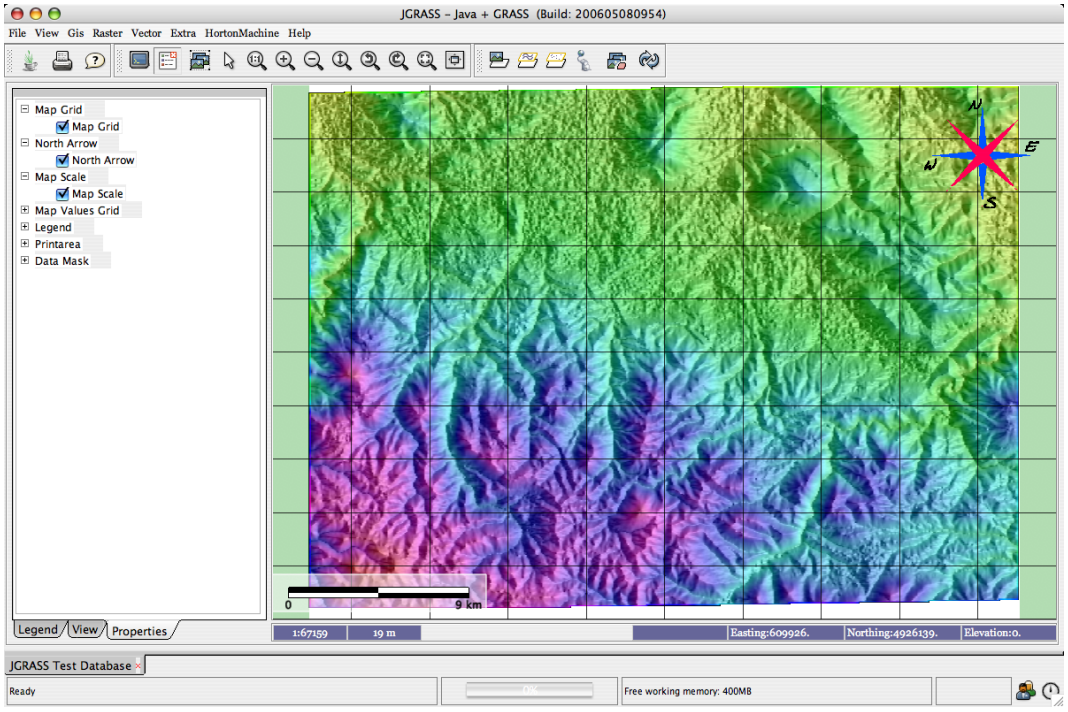

Figure 5.8: *JGrass* map display properties.

- Map scale displayed, the position parameters have been set so as to place it in the lower left part of the screen

- Map values grid: this has not been displayed because the option is not completely activated yet. We advise you to display the map values grid only for small portions of the map at any one time as the operation could require some time.

Map Legend

In order to display the map legend it is not sufficient to check the option that appears in the menu shown in figure 4.24, this option in fact only works after all the parameters necessary for the creation of the map legend have been defined. In order to define these parameters you will need to select *Legend* from the properties list and define its properties. If, for example, you wish to create the map legend for the *elevation.dem* map, you will have to do so by means of the interface shown in figure 5.9.

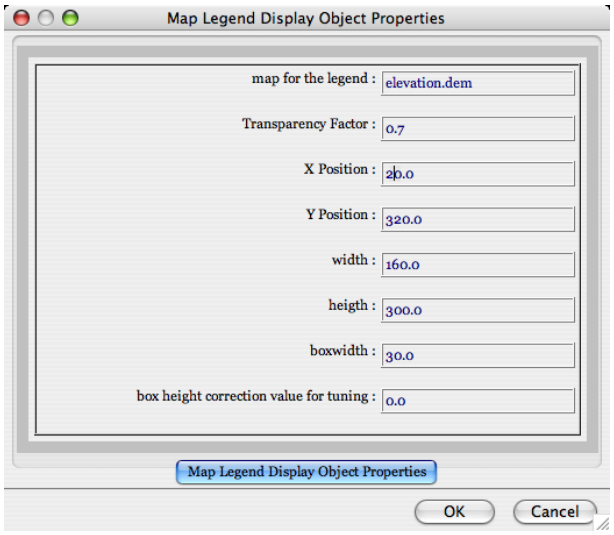

Figure 5.9: Definition of the map legend properties.

It is possible to set the map legend position, its transparency factor, and the size of the box that contains the legend. The legend will vary according to the map to which it refers, not only in its values but also in its form. The legends created for the *elevation.dem* and *geology* maps, shown in figures 5.10 and 5.11 are very different from each other: the former is a "continuous" legend in which the colour scale used in the map is displayed along with some reference values, while the latter is a "discrete" legend where for each specific colour there is an associated value or, in this particular example, the name of the corresponding geological formation. This difference is due to the fact that for the *geology* map the values were categorized and the legend refers to these categories.

Warning: The display of a legend is conditioned by the display of the corresponding map. In

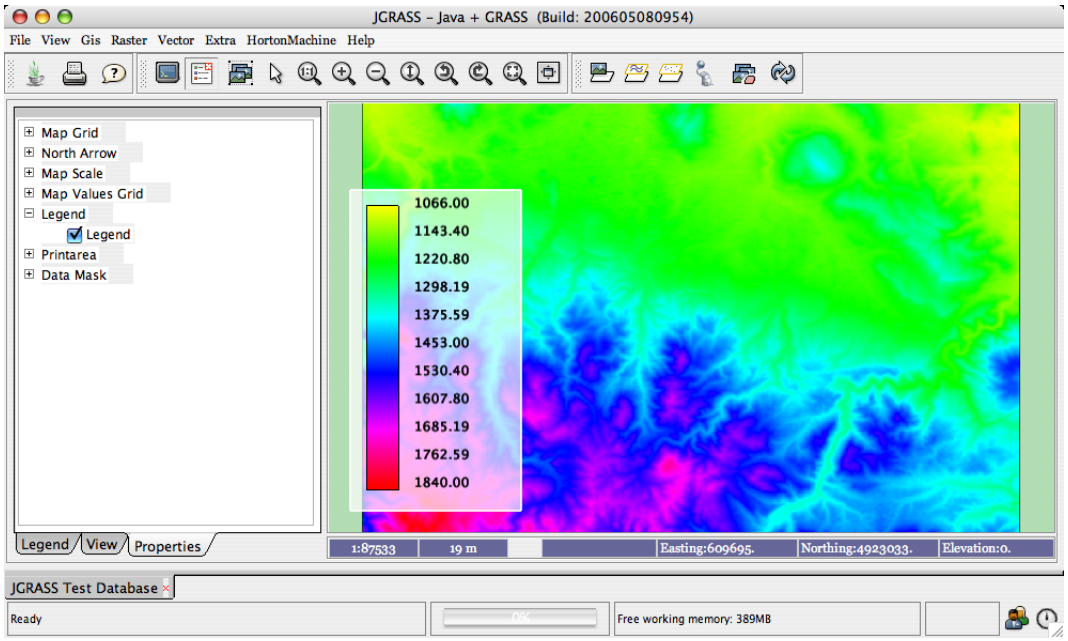

Figure 5.10: *JGrass* view of the elevation map and its legend, created with the parameters shown in figure 5.9.

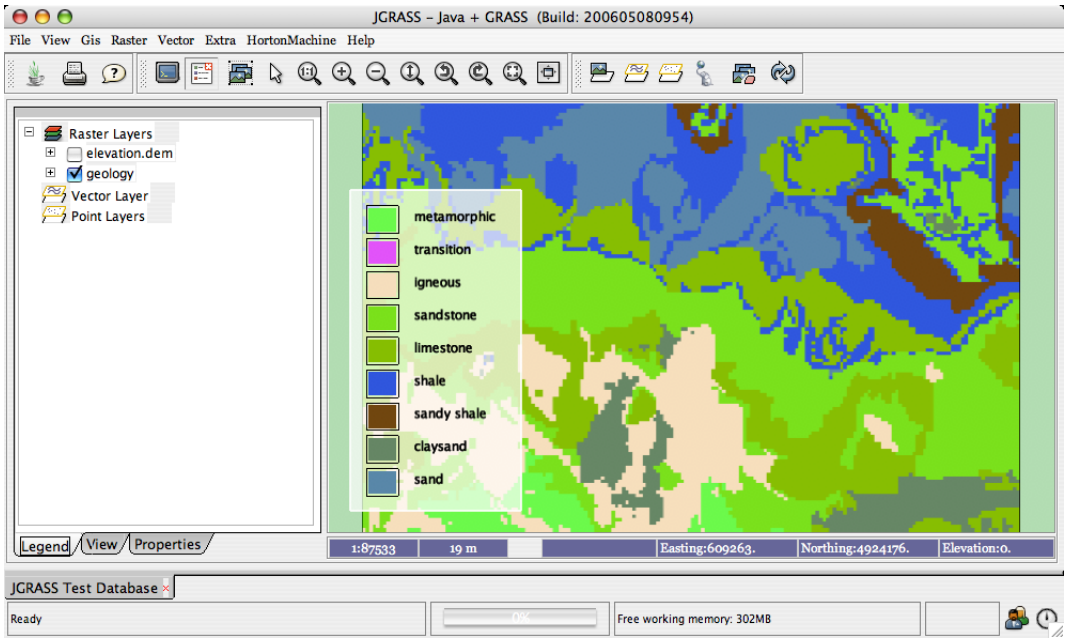

Figure 5.11: *JGrass* view of the geological formations map and its legend.

other words, to display a legend you must also display the corresponding map, along with other maps if the need arises.

5.3.2 Vector Data

By clicking on the icon the vector map display wizard for the visualization of vector maps in shapefile format is launched. The initial window asks you to select the path to the file to be displayed, in fact you can load this type of data from any directory of the filesystem. In the Spearfish dataset a directory called *shapefiles* has been created in the Spearfish location. To view these data, in the dialog box shown in figure 5.12, you will have to select the path to the *JGrass* installation directory and then select `.../spearfish/PERMANENT/shapefiles`, the available shapefiles are here.

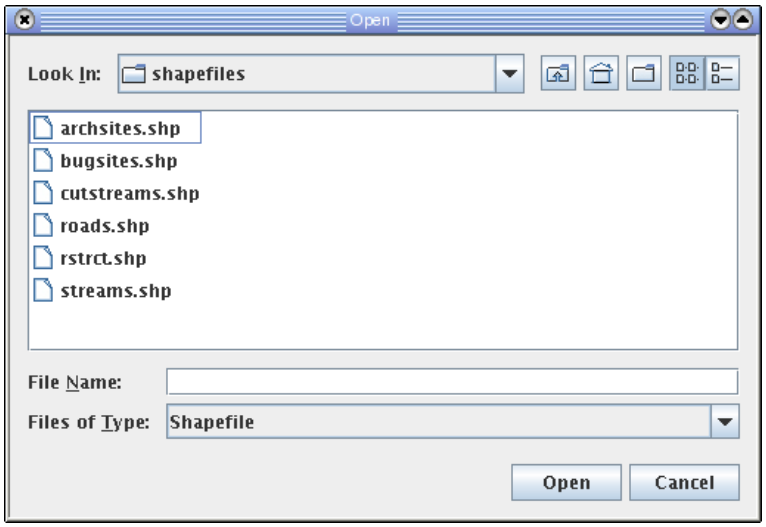

Figure 5.12: Dialog box for the selection of a shapefile to be displayed with *JGrass*.

Display Properties

Repeating the above procedure in order to display the *roads* and *streams* shapefiles you will see that the two maps are displayed as shown in figure 5.13, the default display does not distinguish between the two data types. To distinguish between roads and streams you have to change the display setting of one or both of the maps. This is done in the same way as is done with raster maps (Paragraph 5.3.1) by right-clicking on the name of the map in the legend section of the screen.

There are three active options for vector data:

1. *Remove + "map name"*: to remove the map from the *JGrass* display, this option does not jeopardize the hard copy of the file, as was seen with raster data.

2. *View data*: to view the entire data table associated to the vector map.

3. *Proprieties*: to modify the map display properties.

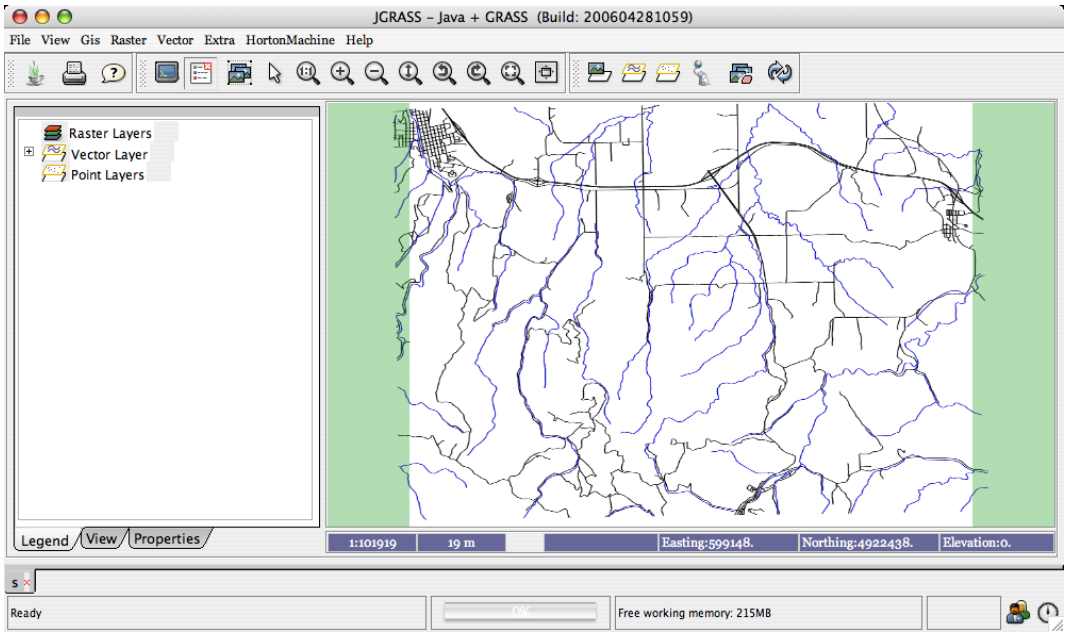

Figure 5.13: View of two shapefiles in *JGrass* with default settings.

By selecting *Properties* the window with the current map display parameters will appear, like the one shown in figure 5.14. This window allows you to edit the settings simply by varying the the number that corresponds to the property you wish to modify, or by clicking on the colour in order to change the display colour. Figure 5.14 also shows the window to choose the display colour. The parameters that can be modified (shown in figure 5.15) vary according to the type of object in the file, in other words it depends on whether you are dealing with polygons (areas) or simply lines and points.

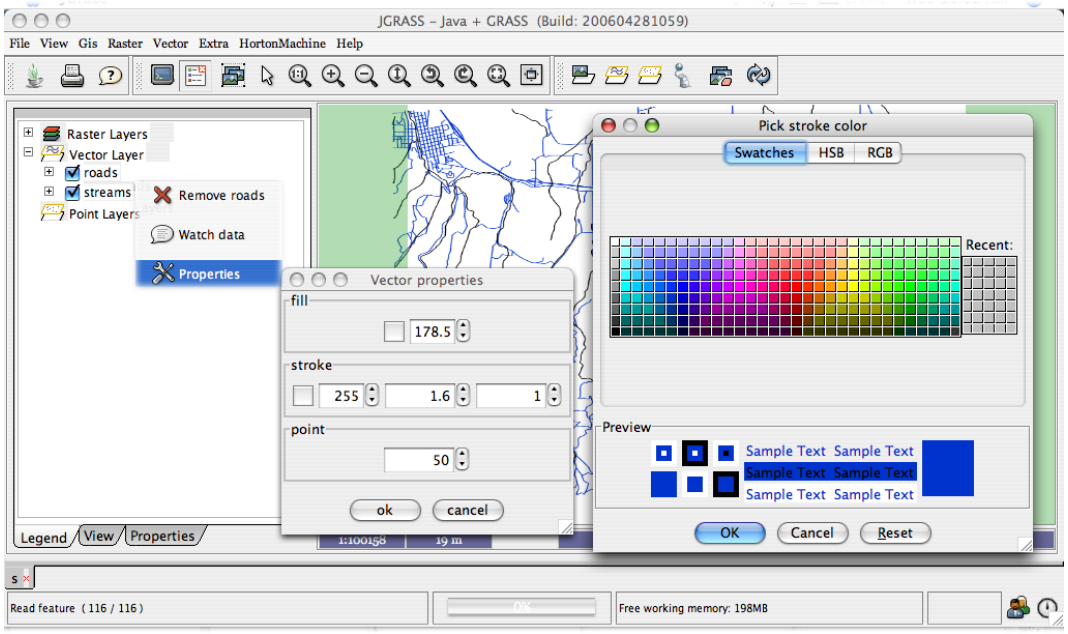

Figure 5.14: Settings for vector map display properties in *JGrass*.

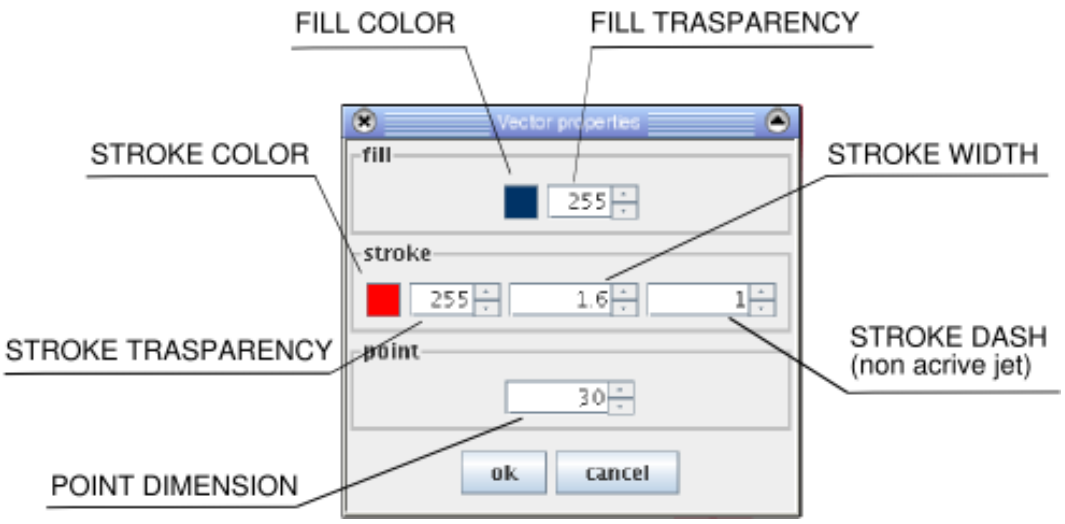

Figure 5.15: Display properties of a vector map in in *JGrass*, distinguished according to the type of object being viewed.

By setting the display colour for the *roads* map to red and setting the stroke thickness to three the map display shown in figure 5.16 is obtained.

If you want to view all the data linked to the loaded map you only have to select the *View data* option from the popup menu. The data are presented in tabular format, for each object of the shapefile it is possible to access its respective attributes. The *roads* shapefile table is shown in figure 5.17. In the example, the attributes are the identification number, ID, and the road category, CAT_ID. As you can see, there are no data in the road description column.

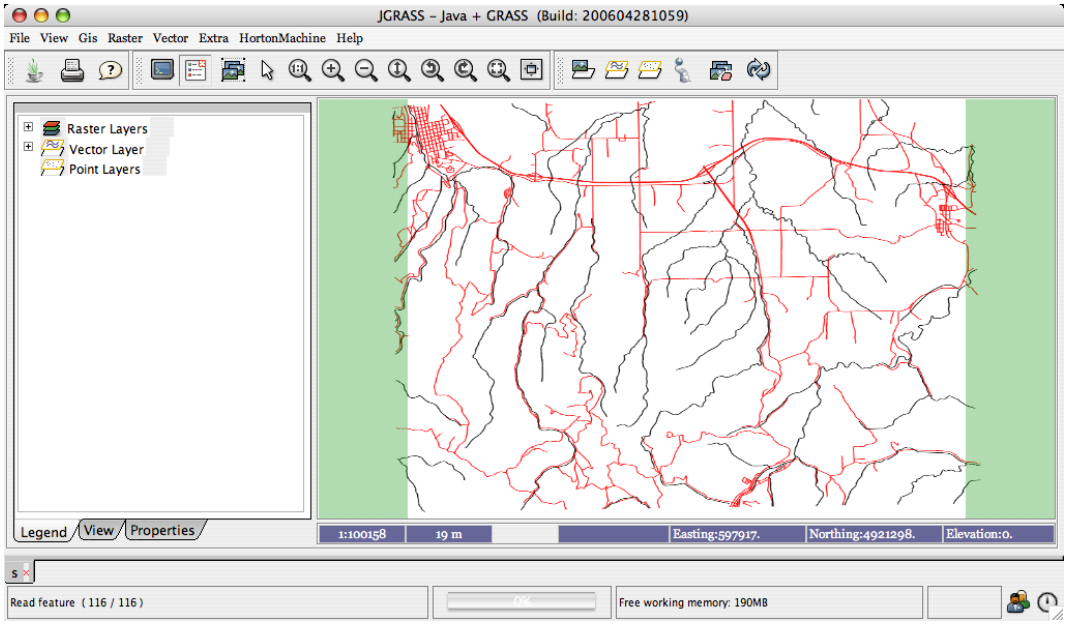

Figure 5.16: View of two shapefiles in *JGrass* with user-customized settings.

Data Viewer

the_geom	ID	CAT_ID	CAT_DESC
MULTILINESTRING ((...	99	1	0
MULTILINESTRING ((...	98	5	0
MULTILINESTRING ((...	97	4	0
MULTILINESTRING ((...	96	4	0
MULTILINESTRING ((...	95	4	0
MULTILINESTRING ((...	94	4	0
MULTILINESTRING ((...	93	4	0
MULTILINESTRING ((...	92	4	0
MULTILINESTRING ((...	91	4	0
MULTILINESTRING ((...	90	3	0
MULTILINESTRING ((...	9	5	0
MULTILINESTRING ((...	89	3	0
MULTILINESTRING ((...	88	3	0
MULTILINESTRING ((...	87	1	0
MULTILINESTRING ((...	86	1	0
MULTILINESTRING ((...	85	3	0
MULTILINESTRING ((...	84	2	0
MULTILINESTRING ((...	83	1	0
MULTILINESTRING ((...	825	3	0
MULTILINESTRING ((...	824	3	0
MULTILINESTRING ((...	823	1	0
MULTILINESTRING ((...	822	1	0
MULTILINESTRING ((...	821	3	0
MULTILINESTRING ((...	820	1	0
MULTILINESTRING ((...	82	1	0
MULTILINESTRING ((...	819	3	0
MULTILINESTRING ((...	818	1	0
MULTILINESTRING ((...	817	4	0
MULTILINESTRING ((...	816	5	0
MULTILINESTRING ((...	815	2	0
MULTILINESTRING ((...	814	1	0
MULTILINESTRING ((...	813	2	0
MULTILINESTRING ((...	812	1	0
MULTILINESTRING ((...	811	1	0
MULTILINESTRING ((...	810	2	0
MULTILINESTRING ((...	81	4	0
MULTILINESTRING ((...	809	1	0
MULTILINESTRING ((...	808	2	0
MULTILINESTRING ((...	807	2	0
MULTILINESTRING ((...	806	2	0
MULTILINESTRING ((...	805	1	0

Figure 5.17: Viewing the attributes linked to a shapefile in *JGrass*.

5.4 Map Query

All the maps viewed in *JGrass*, be they in raster or vector format, can be queried, in other words it is possible to know the value of a specific point of the map.

5.4.1 Querying Raster Maps

The command used to query raster maps is *d.what.rast*, it is in *Raster* menu, in the *Query raster maps* section (see figure 5.18).

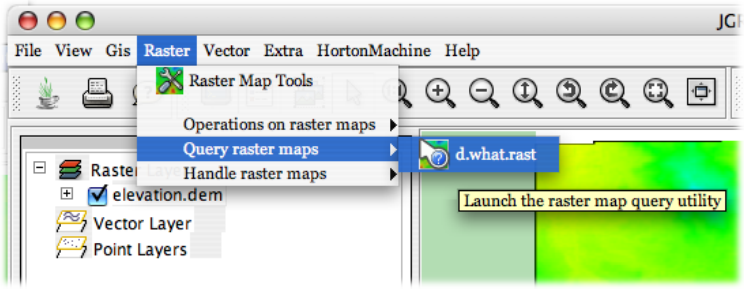

Figure 5.18: The menu to select the query raster map command.

The query raster map command has a handy graphic interface that allows you to choose a map and query it simply by clicking on the point you want to know the value of. With this command you can also query maps that are not displayed and you can query various maps simultaneously by selecting them from the menu of available maps. With each click on the map the value of the selected point will appear, if various maps are queried various values will appear. The value of the previously clicked point remains visible, as well as its coordinates, and the distance between the two points (the previously selected point and the current one). The graphic interface is shown in figure 5.19 where the *elevation.dem* and *geology* are queried simultaneously. To close the command click on the *exit* button.

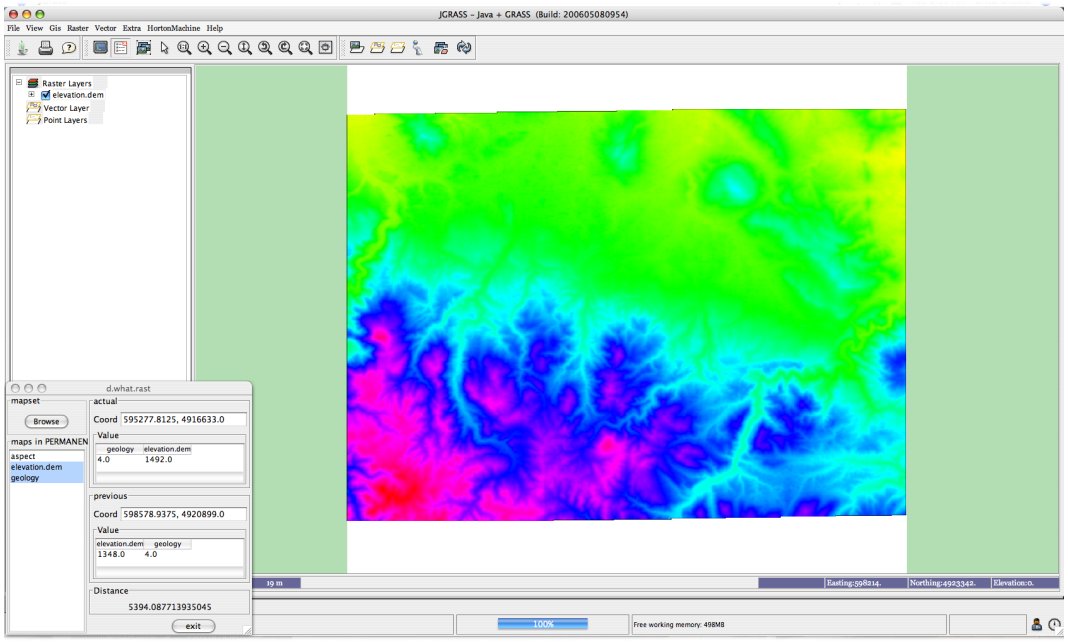

Figure 5.19: Graphic interface for the query raster map command

5.4.2 Querying Vector Maps

The command used to query vector maps is the only vectorial analysis command that is currently integrated in *JGrass*, it is found in the *Vector* menu (as shown in figure 4.14). A graphic interface has also been implemented for the *d.what.vect*, similar to the one seen for the *d.what.rast* command (see figure 5.20), which gives a menu of the available maps (in this case, only the maps that have been loaded as the command queries shapefiles), the coordinates and attributes of the clicked point, and the distance between two successively clicked points.

The only peculiarity of the vector command, with respect to the raster command, is the number in the upper left of the interface which is called *snap*. This represents the maximum distance at which it is possible to click in order to select an object. The *snap* value is specific for every map, indicatively you can set the map resolution as the initial value. If, by clicking in a point, the data relative to two objects appear it means that both objects are less then the *snap* distance from the clicked point. In this case, in order to select one object it is sufficient to reduce the *snap* value.

It is not possible to select various maps with the query vector map command: you can query only one map at a time. To exit the command click the *exit* button.

As a revision of a part of what has been seen in the previous paragraphs you can view the *elevation.dem* map, as in figure 5.21, and modify the map scale and the transparency options of the displayed maps.

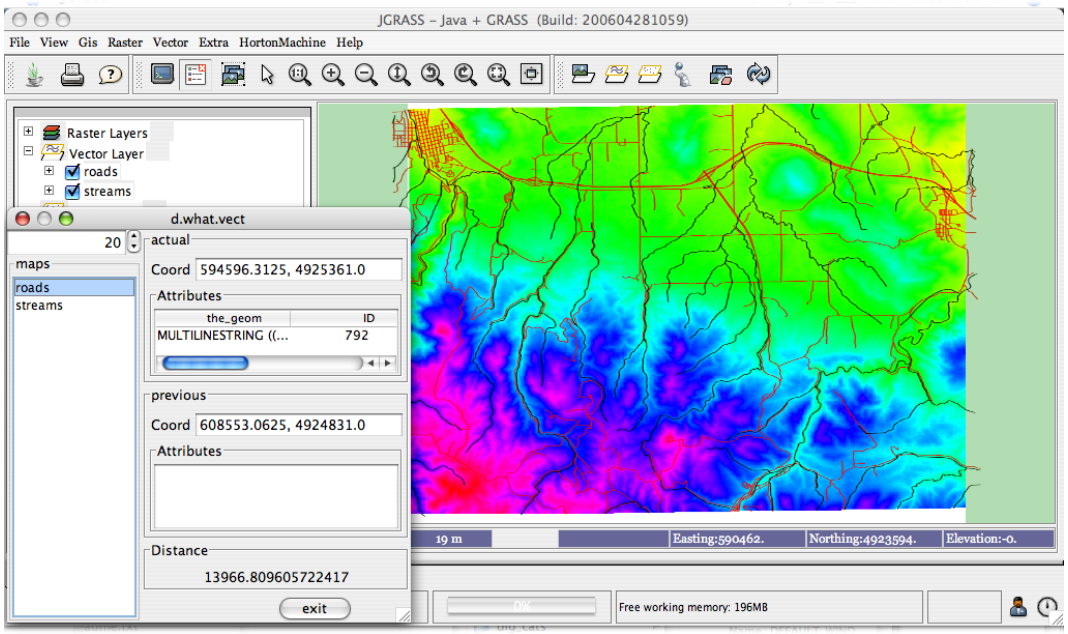

Figure 5.20: Graphic interface for the query vector map command

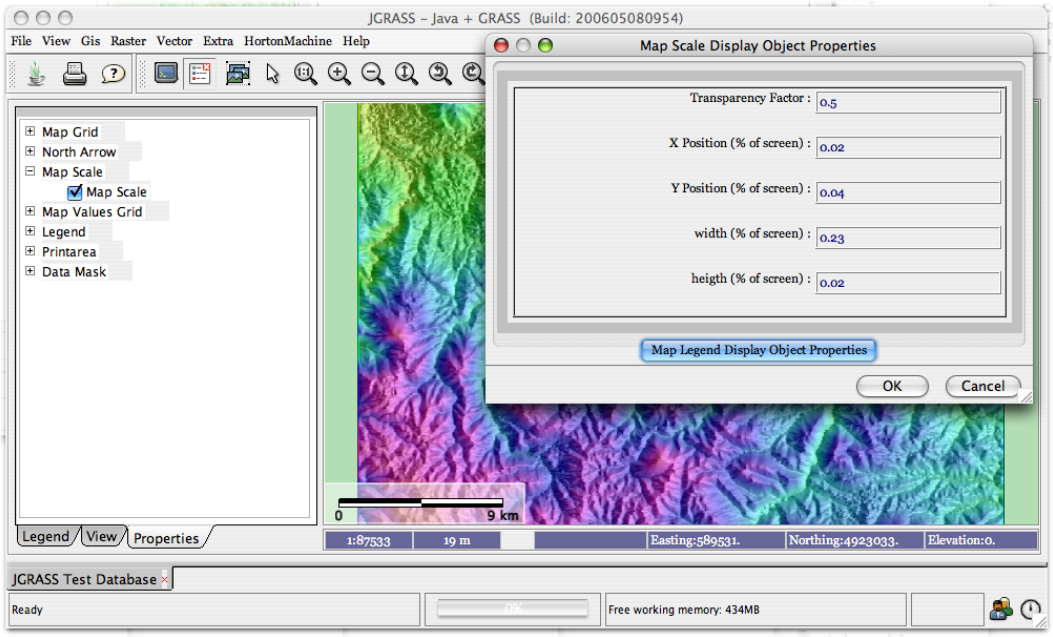

Figure 5.21: *JGrass* view of the map scale with user defined configurations.

5.5 Plotting Map Profiles

In *JGrass* it is possible to plot the profile of a map in a section drawn on-screen by means of the *r.profile* command, found in the *Raster - Operations of raster maps* . The graphic interface is shown in figure 5.22, it has information about the input data and the output data of the command.

The information about the input data is found on the left part of the interface and it concerns the choice of mapset (from those loaded during the workspace creation process) and the map for which a profile is desired (in the example a profile is plotted for the *elevation.dem* map).

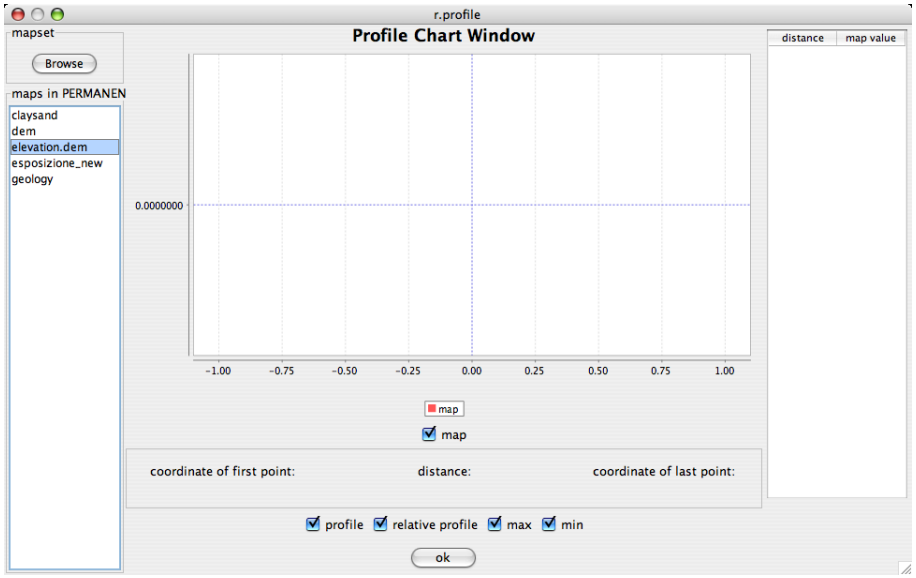

Figure 5.22: Graphic interface for the *JGrass* plot profile command.

At this point you have to move the profile chart window so that the portion of map where you want a profile is visible and plot with the mouse the desired section. The mouse pointer has a different appearance during this operation so that it is clear that you are now in plot mode. It is sufficient to click on the first and last points of the desired cross-section. A blue line will appear on the map, in correspondence of which a profile will be plotted (see figure 5.23). In the profile chart window the information relative to the profile will be displayed, more or less quickly according to the map resolution and the length of the cross-section, as shown in figure 5.24.

The output of the *r.profile* command are, in the graphic part, the profile of the map in correspondence of the selected section with the maximum and minimum marked, and the relative profile. The other information in the profile chart window is numeric:

1. coordinates of the first point of the section
2. coordinates of the last point of the section
3. section length
4. profile data (progressive distance and altitude)

The profile data can be copied directly from the graphic interface and pasted into a spreadsheet. To modify the information displayed graphically it is sufficient to uncheck the check boxes at the

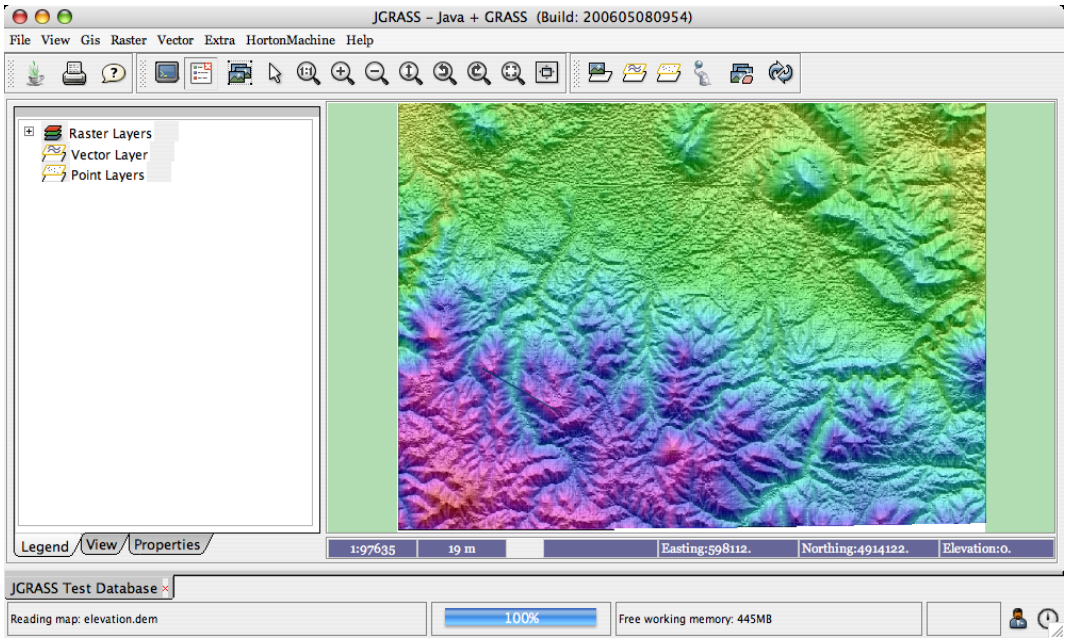

Figure 5.23: Screen view of the section for which a profile is required.

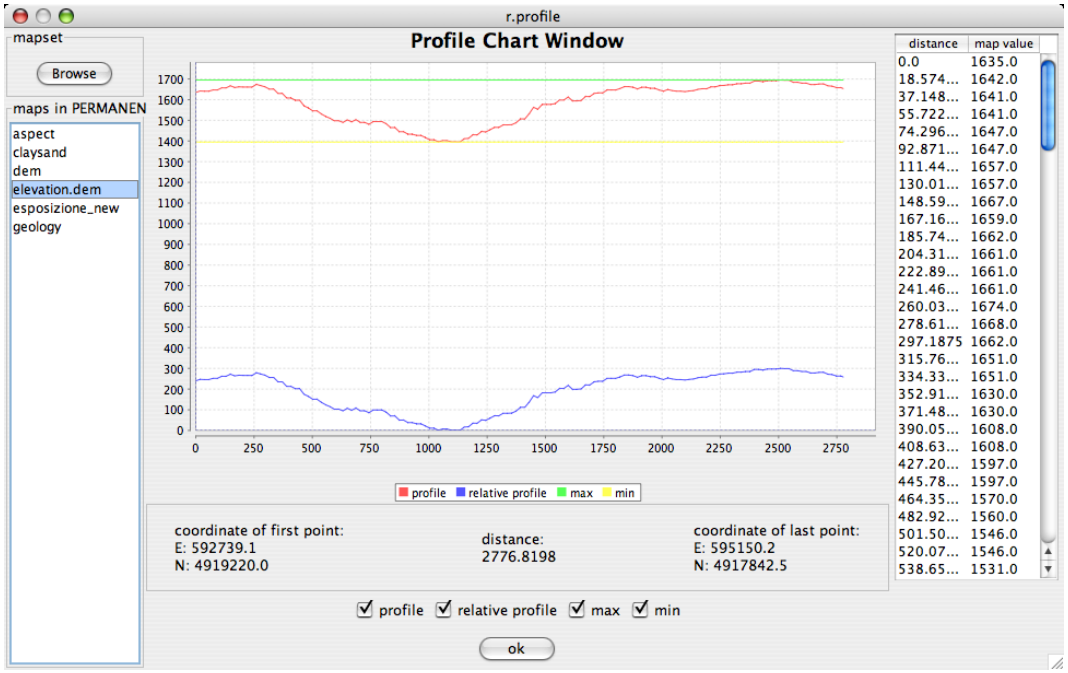

Figure 5.24: Output of the profile plotting command in *JGrass*.

bottom of the profile chart window in correspondence of the information that you do not want displayed.

It is possible to zoom-in to the graph by selecting with the mouse the part of the graph you wish to see better. The selected window will be displayed in the zoom. To return to the initial

view you have to click and drag upwards anywhere within the graph window.

To exit from the profile chart window click on the *ok* button. Leaving the command active while carrying out other operations causes the program to malfunction as a result of continually loading new sections where to plot profiles every time you click on the map. You will notice that upon exiting the command the section that was plotted on the map is still displayed. This has become a new element that can be viewed in the *legend* section of the *map properties window* as `elevation.dem-profile-1`.It can be removed from the display by unchecking the corresponding check box, or it can be completely removed by selecting the *Remove* option from the popup mnnu that appears when you right-click on the map name.

5.6 Handling Raster Map

In the raster manu you will also find the map handling commands, these are the commands that allow you to copy maps from mapsets or particular directories, rename maps, or remove them from the filesystem. All these commands follow a similar logic and their respective graphic interfaces are very similar.

5.6.1 Copying a Map

To copy a map from another mapset or from one of the directories in which the data is stored in the format indicated by the name of the directory the command is *r.copy* . The first panel of the copy raster map wizard asks you to select the directory of origin of the map to copy (by default it is GRASS and the dropdown shows the other directories accessible from *JGrass*). Click *Next* to continue.

Once you have chosen the directory of origin it is possible to choose the map to copy from the list of loaded mapsets. In the example of figure 5.26 the *elevation.dem* map is being copied.

Once you click *finish* a panel similar to the previous one appears where you are asked to choose the destination directory. In the example the map is copied to the *esriasciigrid* directory (see figure 5.27). Click *Next* to continue.

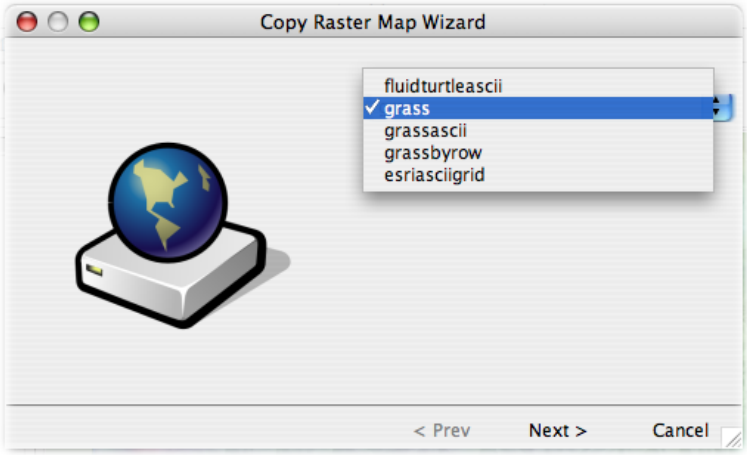

Figure 5.25: First panel of the *JGrass* copy raster map wizard

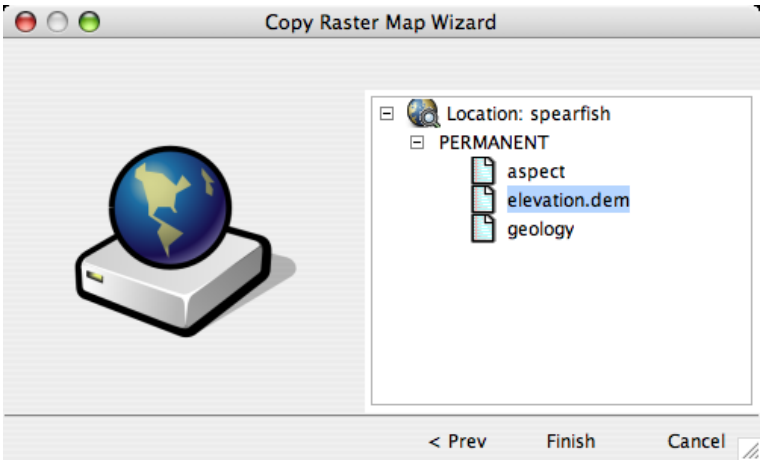

Figure 5.26: Second panel of the *JGrass* copy raster map wizard

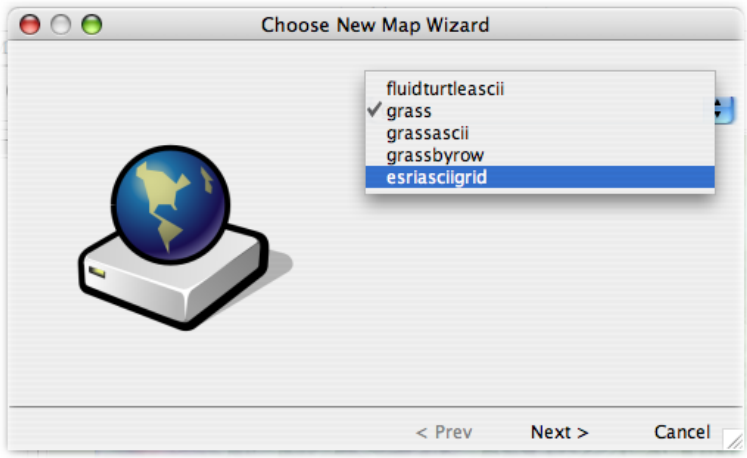

Figure 5.27: Third panel of the *JGrass* copy raster map wizard

In the last panel you are asked to choose the mapset where you want to copy the map and to write the name with which to save it. In the example the only loaded mapset is PERMANENT, therefore once you have selected the mapset and written the new name, *dem*, you can click *Finish* to end the operation. It is very important to select the destination mapset, even if it is the only one listed, otherwise the map will not be copied and the procedure will have to be repeated from the beginning. If the process went well and the map was copied successfully the message, shown in figure 5.29, confirming the creation of the new map appears. Click *ok* to end the operation.

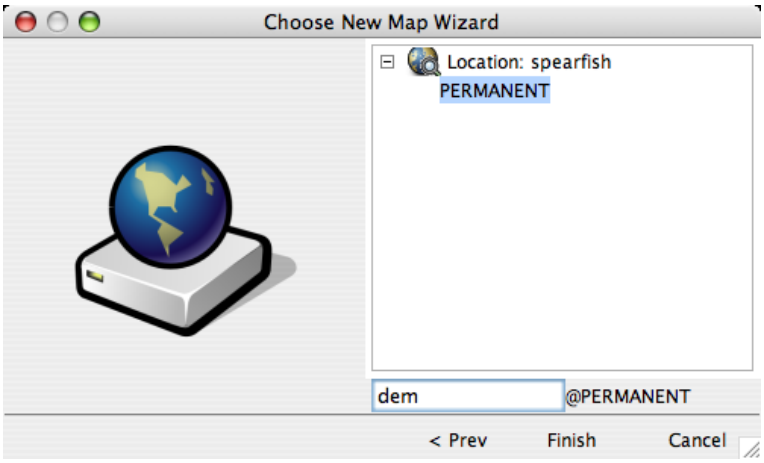

Figure 5.28: Last panel of the *JGrass* copy raster map wizard

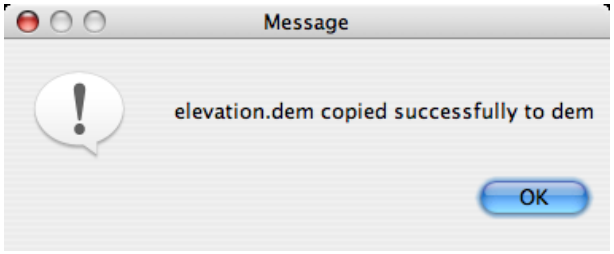

Figure 5.29: Closing message of the *JGrass* copy raster map wizard

The new map can be viewed with the display command or used for the processing and creation of new maps. Remember that the *dem* map is in the *esriasciigrid* directory and is therefore saved in ESRI ASCII format and can be used as an exchange map with other applications.

To consolidate the concept of copying a map from one directory to another we propose that you also copy the other maps of the Spearfish dataset, respectively:

- *aspect* to *grassascii* format with the name *esposizione*
- *geology* to standard GRASS format with the name *geologia*

5.6.2 Renaming a Map

The *r.rename* command is used to rename a map or to move it to another directory or mapset. The rename raster map wizard is very similar to the one for copying maps. The first two panels, where the original map is selected, are the same (see figures 5.25 and 5.26). In this example, however, the newly created *esposizione* map is selected and renamed *esposizione_new* in the *esriasciigrid* directory. The *r.rename* command actually moves the map to the place of the new map specified in the following panel (see figures 5.30 and 5.31) and removes the original map, as opposed to the *r.copy* command that keeps the original map. At the end of the procedure, if all went well, a message appears (see figure 5.32).

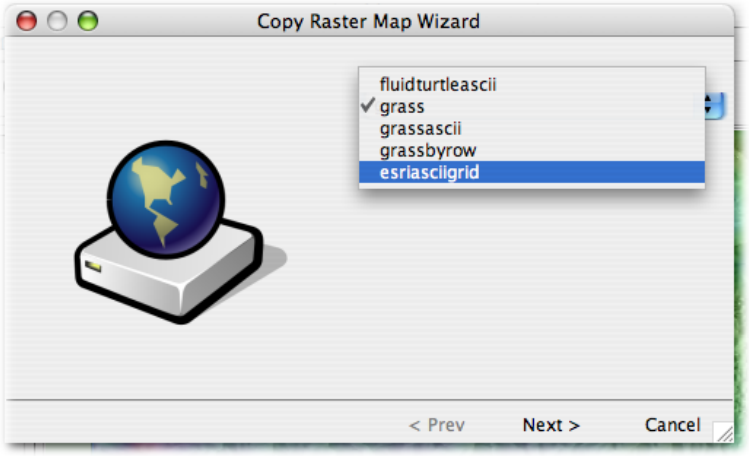

Figure 5.30: Third panel of the *JGrass* rename raster map wizard

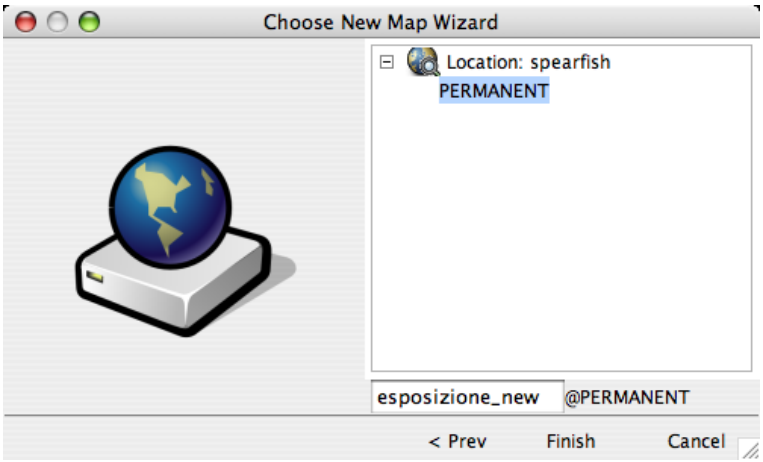

Figure 5.31: Last panel of the *JGrass* rename raster map wizard

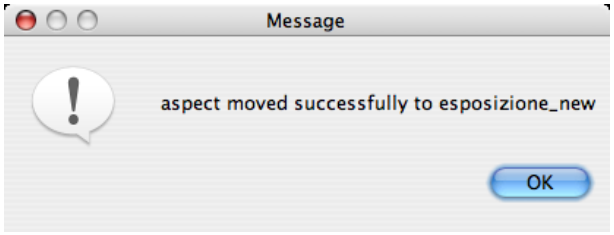

Figure 5.32: Closing message of the *JGrass* copy raster map wizard

5.6.3 Removing a Map

To remove a map from the file system, so that it is no longer available for processing and so free space on the disk, the command to use is *r.remove* , which launches the graphic interface shown in figure 5.33.

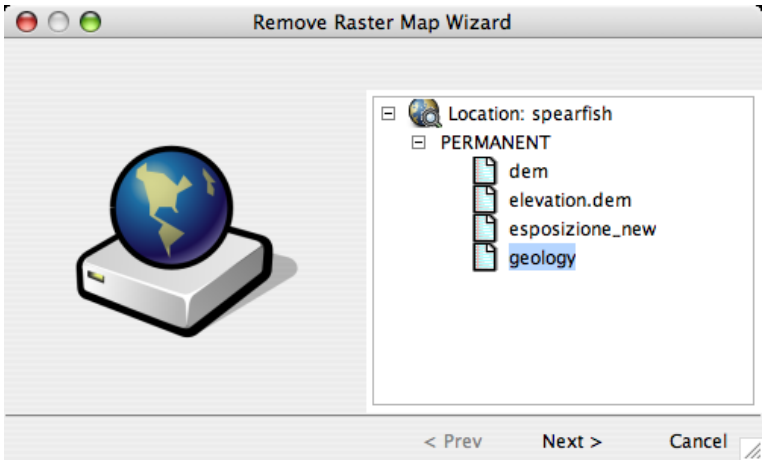

Figure 5.33: First panel of the *JGrass* remove raster map wizard

Figure 5.34: Dialog box for the removal of a map from the filesystem in *JGrass*.

At the moment, the removal of maps is only possible for maps saved in GRASS/JGrass format. If you want to remove the *geology* map, select it from the list and then click *Next* and *Finish* to proceed with the operation. Before removing the the map you are asked to confirm your choice by

means of a dialog box like the one shown in 5.34[1].

5.7 Creating a Colormap for a Raster Map

The raster maps that do not have a colormap associated are displayed with a standard colour scheme that divides the range of values of the map into five subranges which are then assigned colours:

- first range:
- second range:
- third range:
- fourth range:
- fifth range:

The colormaps are found in the *colr* directory of the mapset of the map of interest. To create or edit a map's colormap use the *r.colors* command . The graphic interface of the command is shown in figure 5.35. Here it is possible to select, as in the previous examples, the mapset and the map for colour editing[2]. For the map used in the example, *elevation.dem*, the range of values is $1066 - 1840m\ asl$.

The first thing to do at this point is to load the existing colormap, should this exist, by clicking the *load original colormap* button. If a message appears stating that no colormap was found you can proceed by clicking *ok*, otherwise the colormap will be loaded and displayed in the left part of the screen. In the example, no colormap was found for the *elevation.dem* map, nothing changes in the window and you can proceed to create the colormap that you like.

To create a colormap for a map you must click on the *add rule* button until you have a sufficient number subranges for the data (in the example shown in figure 5.36 it was decided to divide the data into four intervals).

It is now necessary to fill the blank text boxes with the highest and lowest value of the data subrange and assign a starting and a finishing colour to the interval. To load the colours first

[1]Under Windows it is possible to remove a map only if has not been viewed during the current work session due to management problems concerning access to the files in use by the system. Therefore, if you wish to remove a map that has been viewed in the current work session you will first have to restart *JGrass* and then proceed with the removal.

[2]For an as of yet unknown reason you have to select the map name twice in order to view the range of variation of the map data

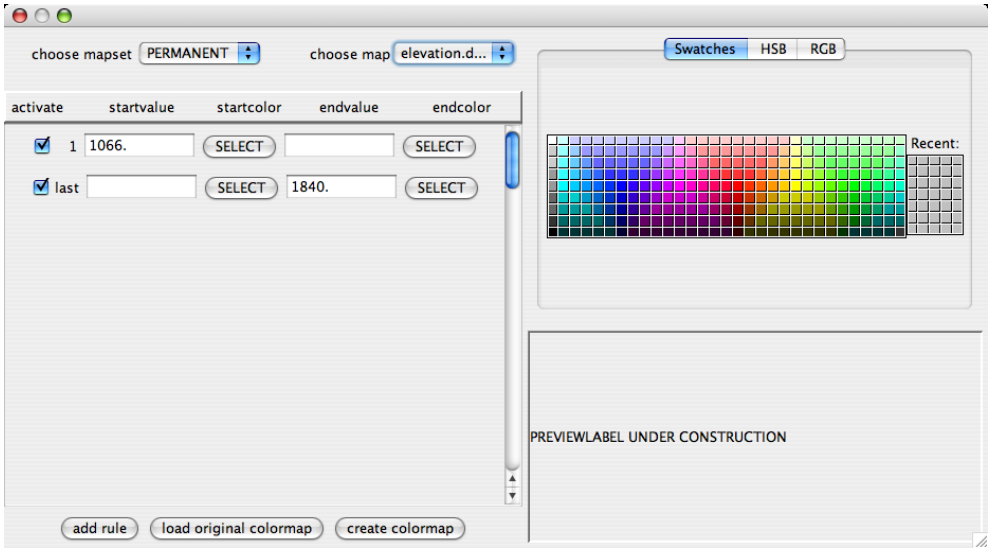

Figure 5.35: Graphic interface for the creation or editing of a raster map colormap in *JGrass*.

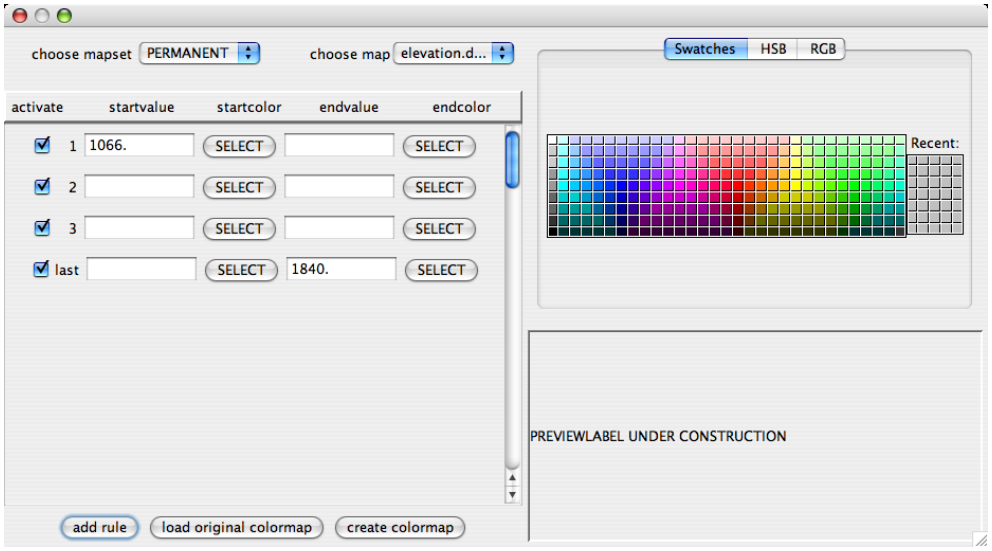

Figure 5.36: Graphic interface for the creation of a raster map colormap in *JGrass* - creation of the data subranges.

you have to select it from the colours window on the right and then click the `SELECT` button in correspondence of where you want to assign that colour. The active colour is also displayed in the `Recent` panel on the far right of the window. Once the colours have been assigned you have to click the *create colormap* button. This is followed by a message confirming the creation of the colormap and indicating the path to where this new file has been saved. To exit the command you simply close the window.

An example of colormap definition is shown in figure 5.37. The graphic results obtained by simply redrawing the map, clicking the icon, is shown in figure 5.38.

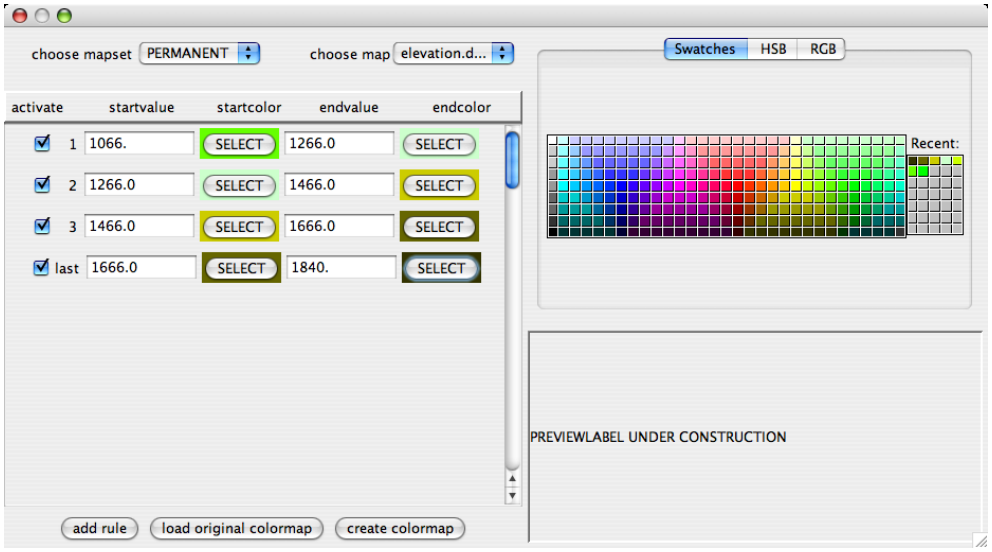

Figure 5.37: Graphic interface for the creation of a raster map colormap in *JGrass* - definition of the subranges and the corresponding colours.

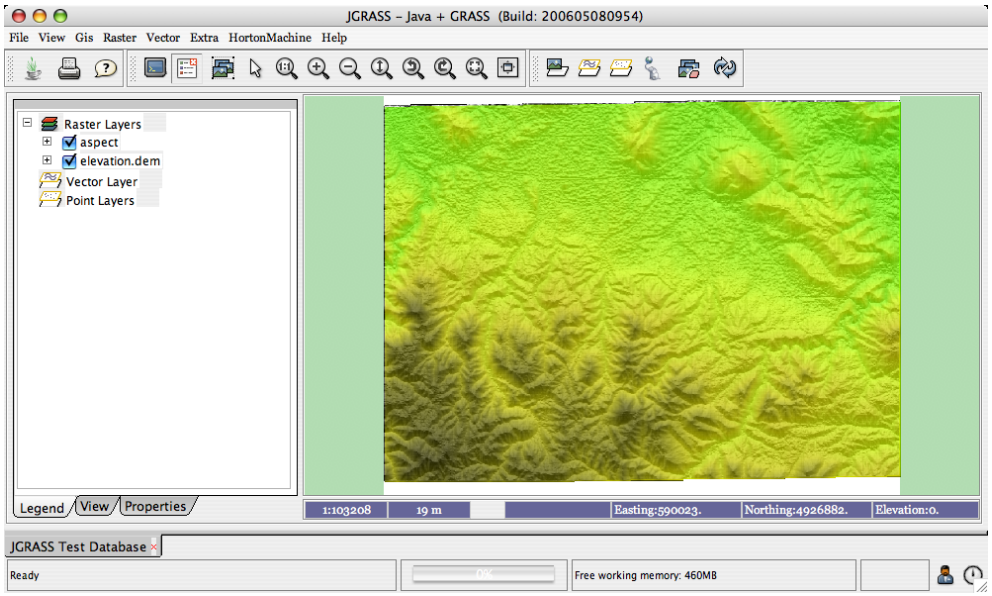

Figure 5.38: The *elevation.dem* map with new colormap.

If the result is not satisfactory all you need to do is recall the colormap creation command,load the colormap created before, and edit everything you do not like, by removing and editing the subranges and by changing the colours. By only changing the colour of the second data interval and recreating the colormap the image shown in figure 5.39 is obtained[3].

[3]If, to display the map with new colormap, redrawing the map is not enough you will have to remove the map from the display (Paragraph 5.3.1) and then redisplay it 5.3.1)

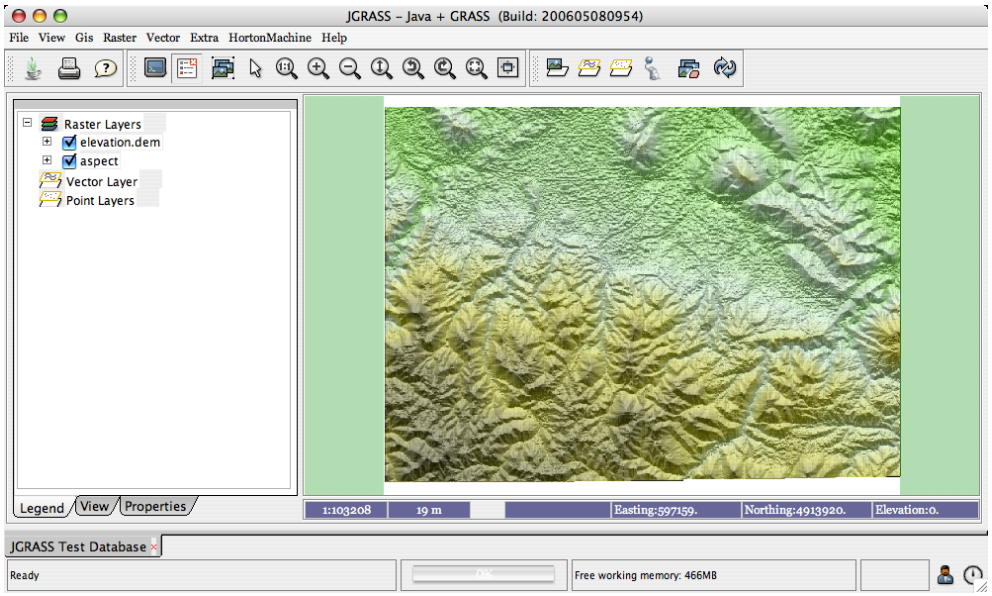

Figure 5.39: The *elevation.dem* map with the modified colormap .

5.8 Calculations with Raster Maps

The command that allows you to carry out mathematical operations of different sorts on raster maps is the *r.mapcalc* command . To carry out operations on a raster means that for every cell of the raster the specified operation is carried out.

The command syntax is very similar to the one used in GRASS but it has a graphic interface which is very easy to use. With *r.mapcalc* you can carry out simple operations such as addition and multiplication of different maps or with values from the same map, you can also carry out more complex operations that rely on trigonometric functions.

The interface of figure 5.40 shows the creation of a new raster,(*prova.dem*), obtained by subtracting $50m$ from the *elevation.dem* map. All operations on raster maps can be carried out by choosing maps from the various mapsets loaded in the workspace and the accessory directories in the *JGrass* location. The choice of these format and mapset varies the list of available maps, which is in the lower left of the window. You can choose a map from the dropdown menu any time you need to insert one in the calculation formula. It is of utmost importance that you specify the name of the map to be created otherwise an error message will appear asking you to fill the missing entries. It is possible to specify the destination for the new map, the default option saves the map in GRASS format in the default mapset, that is shown beside the map name[4]. Clicking *ok* launches the processing operation that can be more or less slow depending on the size and resolution of the active data region. Clicking *cancel* closes the command without carrying out the operation.

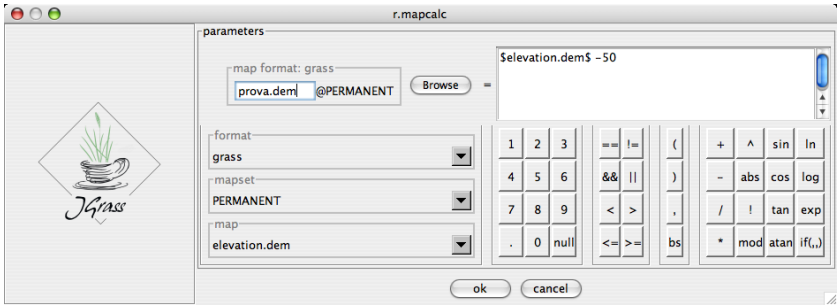

Figure 5.40: Graphic interface for the raster map algebra command.

After the processing you can view the newly created map and check if the calculations are correct by running *d.what.rast* (Paragraph 5.4.1) on the two reference maps, *elevation.dem* and *prova.dem*. The difference of values in every corresponding point of the two rasters should be $50m$.

A very much used operator in raster map algebra is the *if* operator. With this operator it is

[4]At the moment the *r.mapcalc* command allows you to use maps from different mapsets and directories, but all the maps used in one single command must come from the same directory, except for the output raster.

possible to carry out combined calculations on maps considering only specific values of a map.

For example, it might be necessary to mark the limits of areas with a "*claysand*" geological formation, which corresponds to the number eight in the "*geologia*", created earlier, and then extract their altitudes from the DEM (*elevation.dem*).

To reach this result it is necessary to recopy the *geologia* map from the *grassascii* to *geology* in the standard format, or else, copy the *elevation.dem* map to the *grassascii* format. In this example, we will use both maps in standard format. Beware, however, that in recreating the *geology* map the original is cancelled and therefore so is its colormap, the visualization of the map will be different from the initial one.

The command that will allow you to carry out the operation that has been just described is *r.mapcalc*. This tool, borrowed from GRASS is very powerful and a description of it can be found in the GRASS Tutorial[5].

To carry out the desired operation you can use the *r.mapcalc if* operator. For the proper synatax it is sufficient to click the *if* button of the graphic interface and a string like follows will appear

`if(,,)`

which means:

`if(`*condition*`,`*action for true condition*`,`*action for false condition*`)`

To complete the command string you must click where a value is to be input and then select a map or write a condition/action.

We advise you to always use the *if* structure, even if you are operating on a single map with the *null* value for external points. This is because some functions, especially the trigonometric ones, do not read the values corresponding to *null*as non-valid values and they therefore calculate data even for points that are external to the map but inside the active data region (the area that usually remains white during map visualization).

The process of creating a new *claysand* raster map, obtained by isolating points in the *geology* map and associating the corresponding altitude from the *elevation.dem* map to them, is shown in figure 5.41. The resulting map is shown in figure 5.42, where the the original *geology* map is displayed with a 0.2 transparency factor while the new *claysand* is displayed with a 0.8 transparency factor.

[5] http://www.gdf-hannover.de/media.php?id=0&lg=en

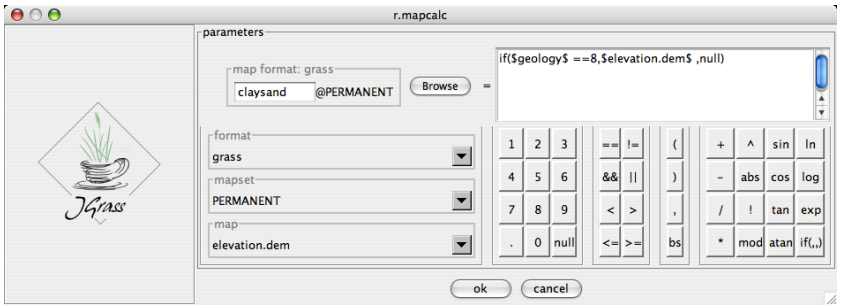

Figure 5.41: Use of the *if* structure during raster algebra

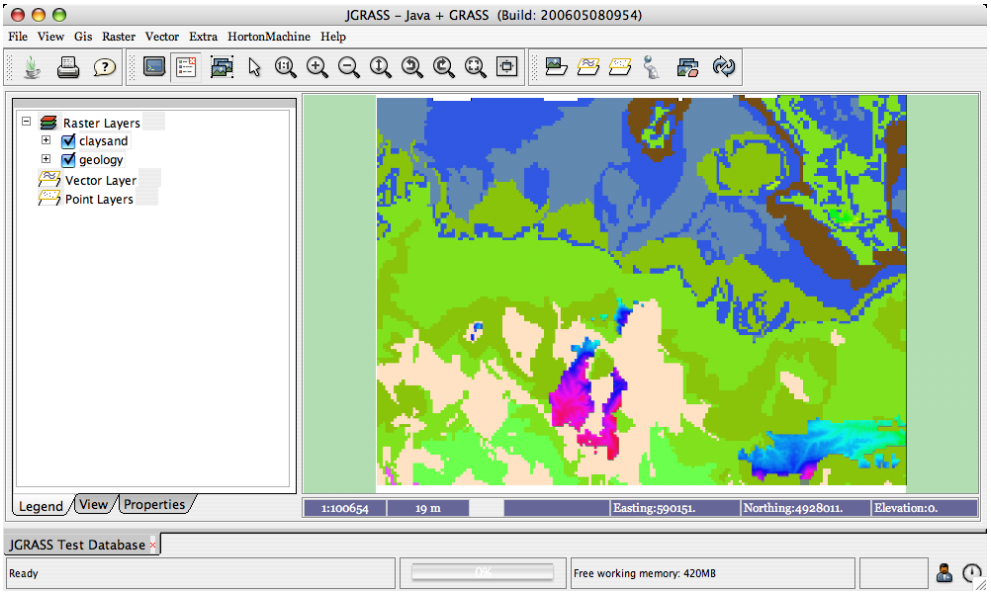

Figure 5.42: View of a map created with the raster map calculation tool.

The *null* value, introduced in the calculation syntax of the *claysand* map, is the value corresponding to NODATA, in other words it is the value to be given to points outside of the area of interest so that they are not considered valid raster cells.

5.9 Creating a Category Table

The categories list allows you to associate a description to specific raster map values. The categories are created with the *r.cats* command .

Let us suppose that you want to recreate the categories for the original *geology* map. What you will need to input in the command window is summarized in table 5.1.

The initial graphic interface is the one shown in figure 5.43, where the *geology* map has already been selected for the creation of a category table.

VALUE	DESCRIPTION
1	metamorphic
2	transition
3	igneous
4	sandstone
5	limestone
6	shale
7	sandy shale
8	claysand
9	sand

Table 5.1: Summary of the creation of the categories table for the *geology* map.

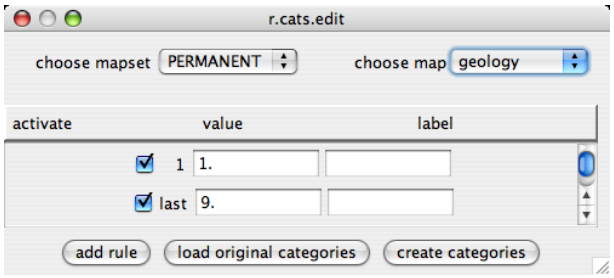

Figure 5.43: Graphic interface of the category editing tool in *JGrass*.

Here, just as with the colormap tool, you must select the map twice in order to load its range of values, then *load original categories*. If the map does not have a category table the category editing window does not change with this operation and you can proceed with the creation of new category table. First of all you must fix the category intervals, in the example we want to assign a description to each value of the map, therefore you must create 9 interval boxes and insert what is listed in table 5.1 (see figure 5.44).

Once the categories have been inserted all you have to do is click the *create categories* button to create the table that has just been defined. If the procedure goes well a message appears confirming the creation of the categories table in the *cats* directory of the current mapset. You can now exit the tool, by simply closing the dialog window, and view the map and legend to check if the categories are recognised by *JGrass* (as shown in figure 5.11).

The *r.cats* tool only works with discrete maps, if you wish to assign categories to the *aspect* map, which is a continuous map, you would first have to create a new raster with a value for every representative interval of original data and only then assign categories. As you might know, the values of the aspect map represent the angle that the gradient forms with respect to the geographical East, therefore:

- NORTH - 90^o

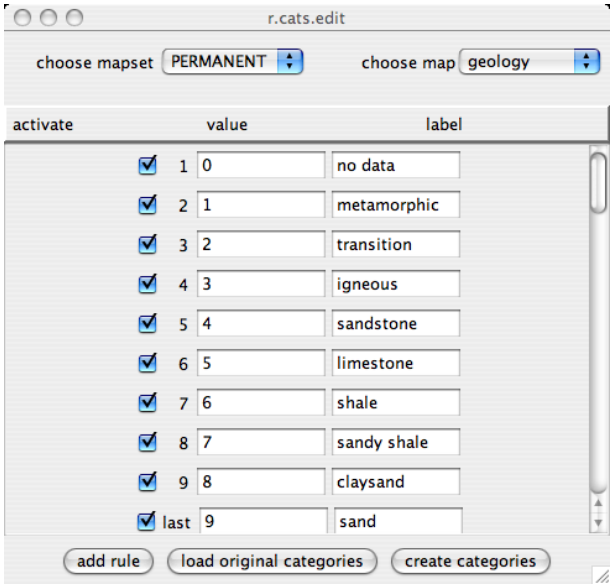

Figure 5.44: Creation of the categories list for the *geology* map in *JGrass*.

- WEST - 180^o
- SOUTH - 270^o
- EAST - 360^o or 0^o

By introducing the intermediate directions (NE - NW - SW - SE) we can divide the map into the intervals listed in table 5.2. The steps to solve this problem are two:

1. create a new aspect map, *aspect.new*, that has the values assigned in table 5.2 (figure 5.45) rather than the original aspect values in degrees, using the raster map algebra tool *r.mapcalc*;
2. create the category table for the new map and associate the legend of the new map to the original map using the category editing tool *r.cats*.

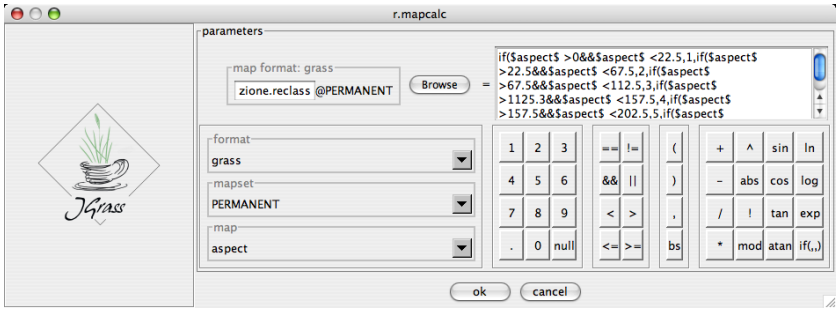

Figure 5.45: Command to create a new raster map based on the *aspect* map but with data from table 5.2.

CATEGORY	NUMBER	INTERVAL
EAST	1	0 - 22.5
NORTH EAST	2	22.5 - 67.5
NORTH	3	67.5 - 112.5
NORTH WEST	4	112.5 - 157.5
WEST	5	157.5 - 202.5
SOUTH WEST	6	202.5 - 247.5
SOUTH	7	247.5 - 292.5
SOUTH EAST	8	292.5 - 337.5
EAST	9	337.5 - 360

Table 5.2: table with the aspect angle values for the creation of the map categories.

5.10 Editing Raster Map Values

Sometimes it is necessary to edit in some way or another raster maps for the most diverse of reasons. If you should need to do so the tool to use is *r.correct*. Actually there are two versions of this tool:

- *r.correct* : allows you to edit all the cells of a raster map one by one;
- *r.correct.polygon* : allows you draw an area on the map and assign a specific value to the entire area.

5.10.1 Editing Raster Maps Manually

The tool to manually edit a raster, *r.correct*, allows you to view the data of a map selected from a list of available maps in a mapset and to edit them manually. The tool, shown in figure 5.46, refers to a portion of the *elevation.dem* map. Using the standard tool options it is only possible to view an area of five cells by five cells, horizontally and vertically, around the cell selected with the mouse. To change the number of cells displayed in the graphic interface you will have to edit the command file `.../etc/prop/r_correct_it.properties`, and in the second line change the five for greater uneven number.

You can edit some values in the window and then click on another part of the map to open another window and so on. Before exiting you can save all the changes in a single saving. You can save the edited map with a new name or keep the original one. Once all the parameters have been set and the editing operations are complete click the *Accept Changes* button to create the new map.

With this tool you can paste some data copied from a window of the same size created in a

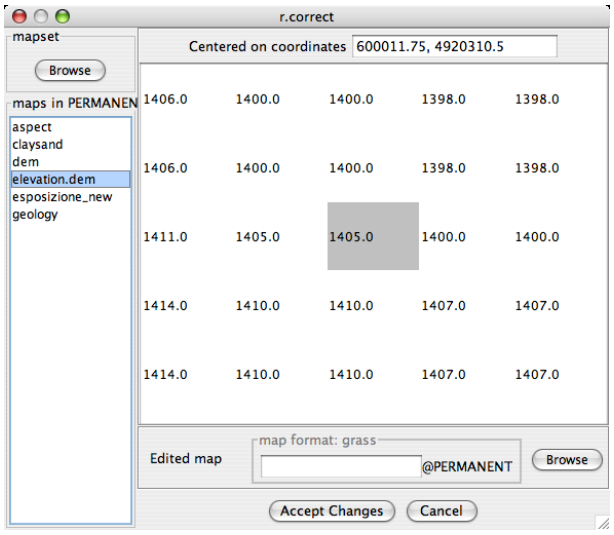

Figure 5.46: Graphic interface of the interactive raster editing tool in *JGrass*.

spreadsheet and so speed up the operation if you have to assign known values to an entire area.

5.10.2 Editing Raster Maps with the Use of Polygons

When you need to edit large areas of a map, for example if you need to exclude areas of a catchment area that drain artificially into other catchments, the *r.correct.polygon* tool is very useful. With this tool you plot a polygon on the map to be edited and so delimit an area to which to assign a specific value, even the *null* value (in fact, in the example of figure 5.47 the *null* is assigned to the area contained within the polygon).

The plotted polygon is visible in blue in figure 5.48.

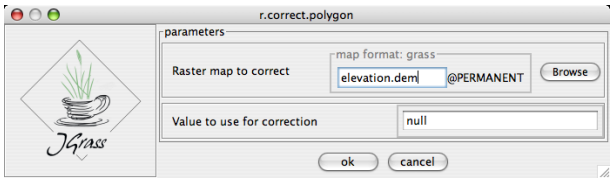

Figure 5.47: Graphic interface for raster map editing with the use of polygons in *JGrass*.

To create a new map it is necessary to click *ok* after having plotted the polygon to start the processing. The exact map display shown in figure 5.49 was obtained by removing the plotted polygon from the menu options in the *legend* panel of the main screen.

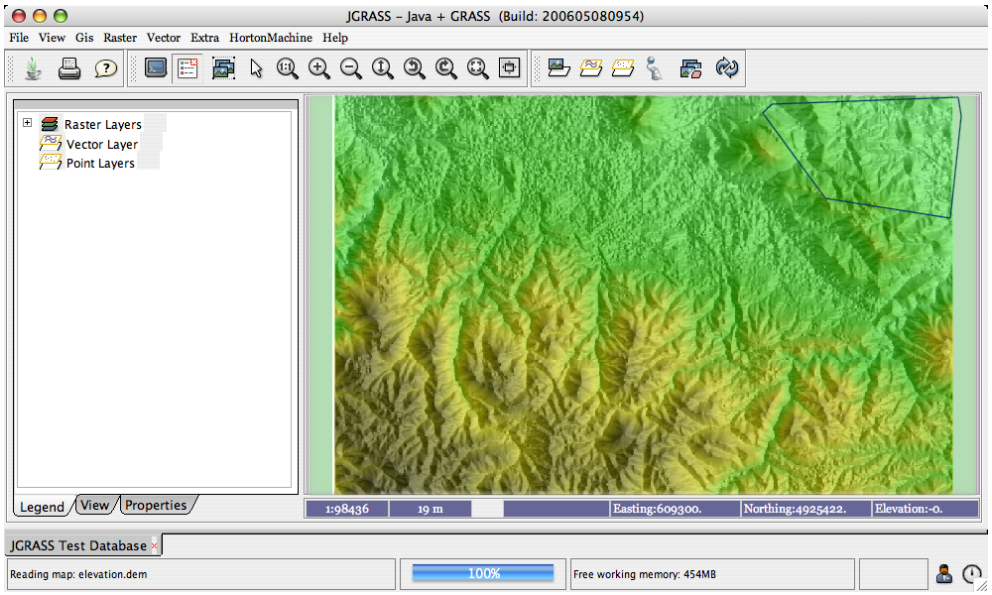

Figure 5.48: The delimiting polygon to which the *null* value will be assigned.

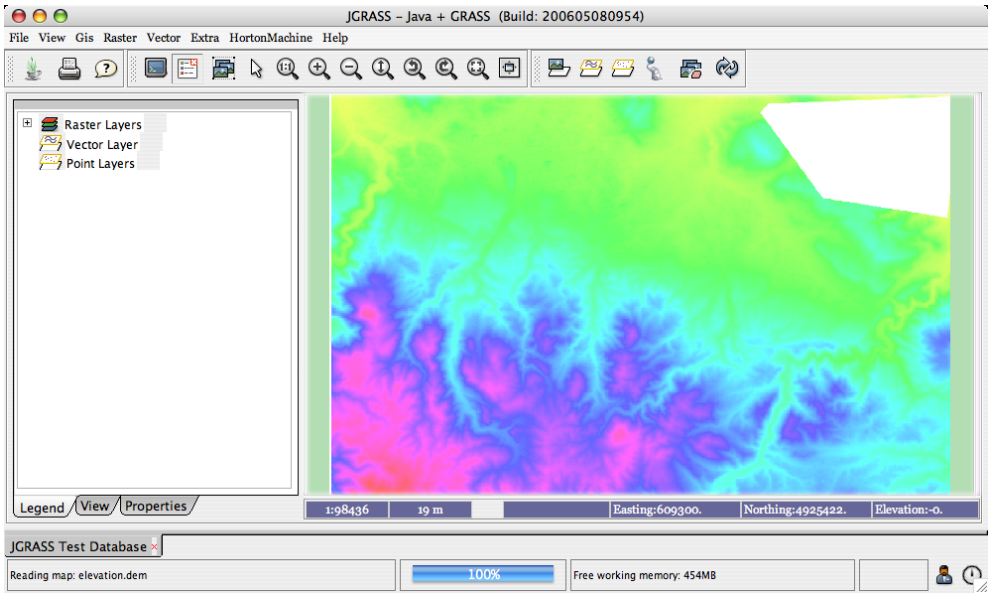

Figure 5.49: Raster map created with the *r.correct.polygon* tool.

6 Creation of a New Workspace

6.1 Creation of a Location

To create a new workspace you have to select the *New Workspace* command from the *File* menu, as shown in figure 4.6. Actually, what in *JGrass* is referred to as a *workspace* is the GRASS *location* concept (Paragraph 3.5). A *location* is the reference directory for all the data and work settings of a project in a specific *JGrass* workspace, its structure is that defined in Paragraph 3.5.

Once you have selected the option to create a new workspace a presentation window appears (Figure 6.1), just click *Next* to continue.

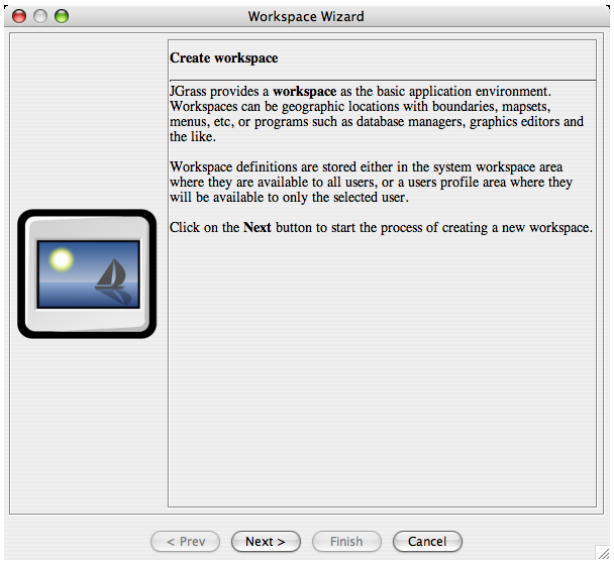
Workspace Wizard
Create workspace
JGrass provides a workspace as the basic application environment. Workspaces can be geographic locations with boundaries, mapsets, menus, etc, or programs such as database managers, graphics editors and the like.
Workspace definitions are stored either in the system workspace area where they are available to all users, or a users profile area where they will be available to only the selected user.
Click on the Next button to start the process of creating a new workspace.
< Prev
Next >
Finish
Cancel

Figure 6.1: Presentation window for the creation of a location.

The following window gives the option to create a workspace compatible with GRASS. The state of the art only permits you to create GRASS location/mapset type workspaces, the development of a data management alternative is currently being developed. You have only to select the `GRASS location workspace` option from the menu and click on *Next*.

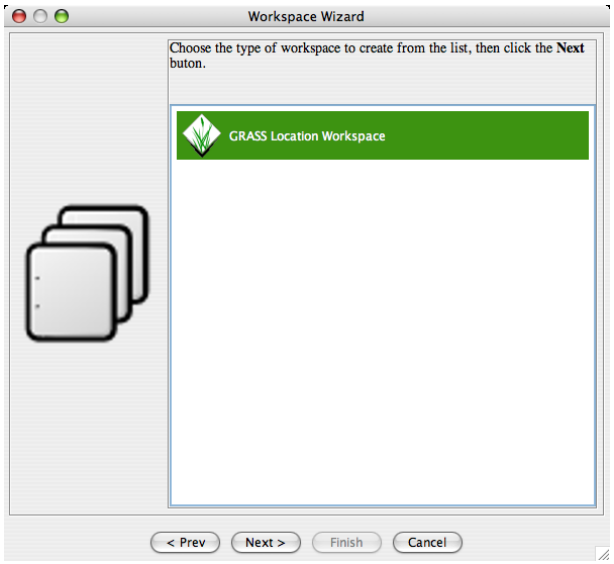
Workspace Wizard
Choose the type of workspace to create from the list, then click the Next buton.
GRASS Location Workspace
< Prev
Next >
Finish
Cancel

Figure 6.2: Window to select the type of workspace to create.

At this point a dialog box will appear where you must specify the path where the new workspace is to be created. The destination path for the new workspace is not simply the GRASS DATABASE but must be complete and contain the name of the new *location*.

If the pathname is not complete or a directory that is not valid is selected, either because its empty or has other data, the warning message shown in figure 6.3 appears. This does not happen if you select an existing location, but in this case you are not creating a new location but rather you are loading a new workspace from an existing location (Paragraph 6.1.1).

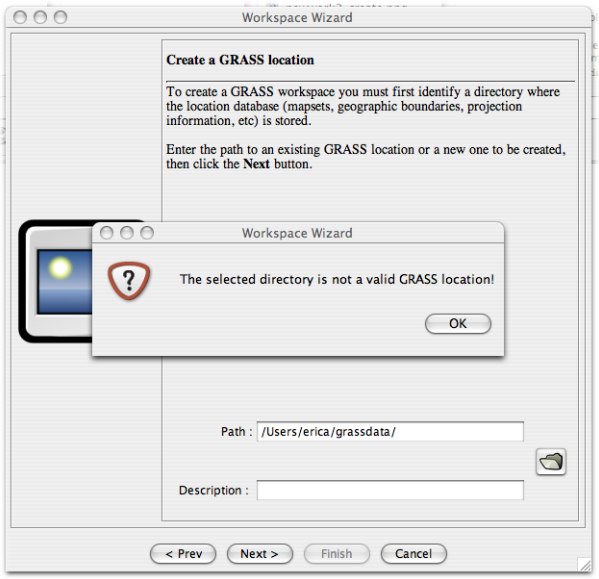
Workspace Wizard
Create a GRASS location
To create a GRASS workspace you must first identify a directory where the location database (mapsets, geographic boundaries, projection information, etc) is stored.
Enter the path to an existing GRASS location or a new one to be created, then click the Next button.
Workspace Wizard
The selected directory is not a valid GRASS location!
OK
Path : /Users/erica/grassdata/
Description :
< Prev
Next >
Finish
Cancel

Figure 6.3: Warning message that the selected pathname is either incomplete or it exists already but it is not valid for saving data from the workspace that is being created.

Once a valid path has been defined it is possible to give a brief description to the workspace (Figure 6.4) and then continue with the operation by clicking on the *Next* button.

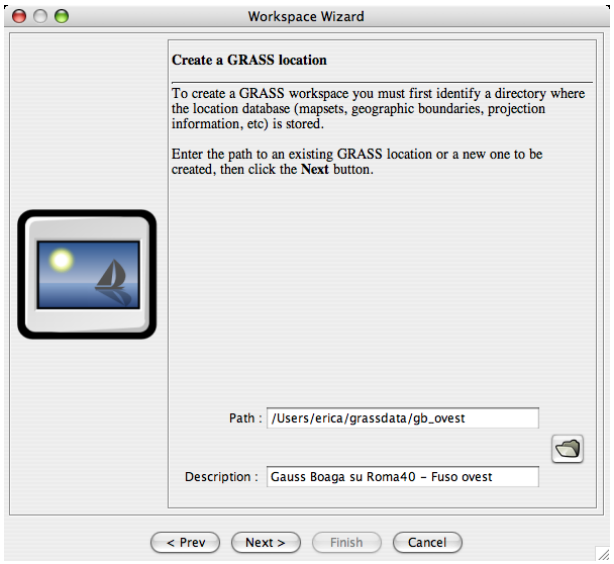
Workspace Wizard
Create a GRASS location
To create a GRASS workspace you must first identify a directory where the location database (mapsets, geographic boundaries, projection information, etc) is stored.
Enter the path to an existing GRASS location or a new one to be created, then click the Next button.
Path : /Users/erica/grassdata/gb_ovest
Description : Gauss Boaga su Roma40 - Fuso ovest
< Prev
Next >
Finish
Cancel

Figure 6.4: Window to select the path where to create the new workspace.

When you create a new workspace you will be warned that a new directory will be created on the filesystem, you only have to *ok* the message to continue.

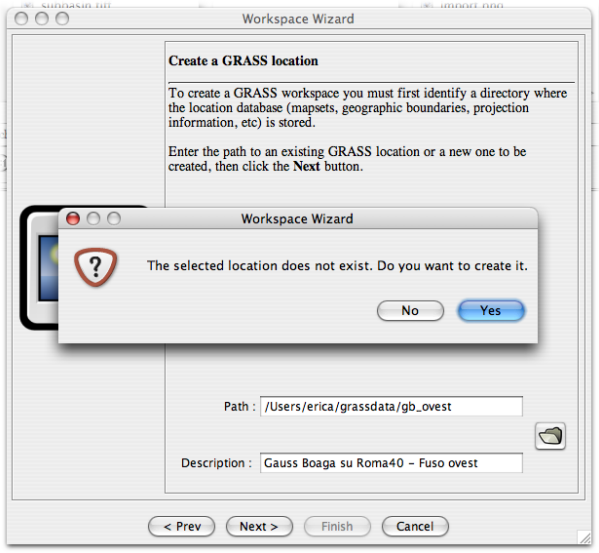
Workspace Wizard
Create a GRASS location
To create a GRASS workspace you must first identify a directory where the location database (mapsets, geographic boundaries, projection information, etc) is stored.
Enter the path to an existing GRASS location or a new one to be created, then click the Next button.
Workspace Wizard
The selected location does not exist. Do you want to create it.
No
Yes
Path : /Users/erica/grassdata/gb_ovest
Description : Gauss Boaga su Roma40 - Fuso ovest
< Prev
Next >
Finish
Cancel

Figure 6.5: Warning message that a new directory will be created in the filesystem.

In order to load a new workspace you need to specify the work region settings, starting with the reference system and the the cartographic projection. If you are unsure of the exact wording to use it is sufficient to write the initials of the country to which the data refers and a selection of the projections to use will be made automatically. All you have to do then is select the correct option from the options proposed with the aid of the notes that appear by passing over the name of a projection with the mouse pointer, as shown in figure 6.6. In the example, it is proposed to create a location using the Gauss Boaga zone West projection.

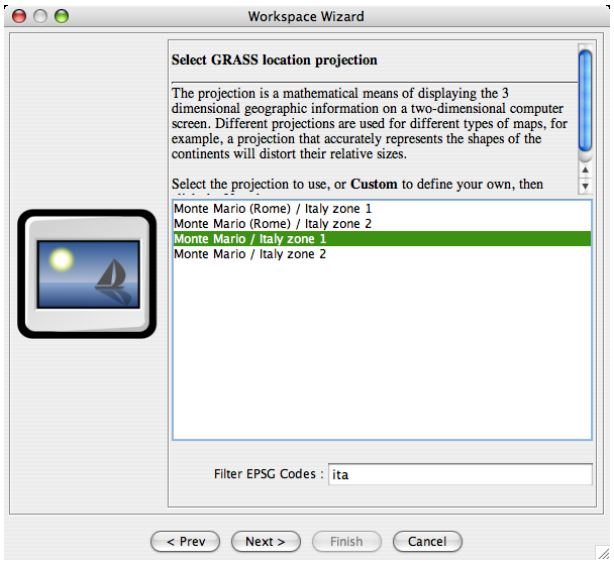
Workspace Wizard
Select GRASS location projection
The projection is a mathematical means of displaying the 3 dimensional geographic information on a two-dimensional computer screen. Different projections are used for different types of maps, for example, a projection that accurately represents the shapes of the continents will distort their relative sizes.
Select the projection to use, or Custom to define your own, then
Monte Mario (Rome) / Italy zone 1
Monte Mario (Rome) / Italy zone 2
Monte Mario / Italy zone 1
Monte Mario / Italy zone 2
Filter EPSG Codes : ita
< Prev
Next >
Finish
Cancel

Figure 6.6: Window to select the reference system and cartographic projection parameters for the new workspace.

The next window has a summary of the the reference system and cartographic projection pa-

rameters selected. All the values are already set to default values, the only ones that we advise you to edit, in order to better manage the workspace, are those referring to the region limits and spatial resolution.

For example, the parameters of figure 6.7 are characteristic of the Trentino region in Italy. To continue just click *Next*.

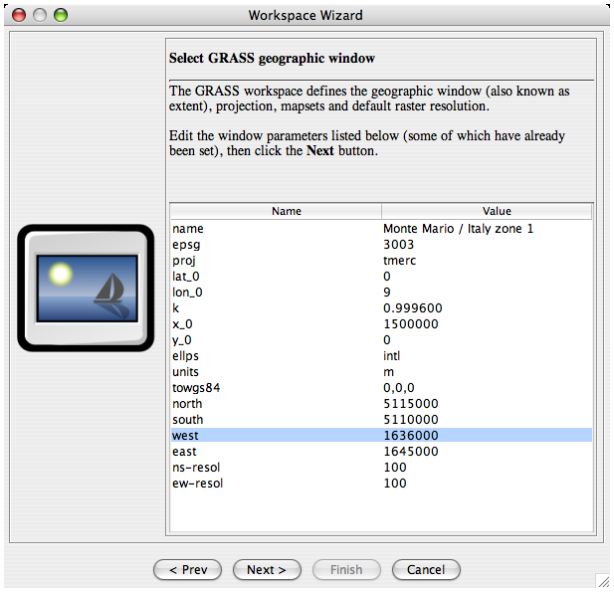

Figure 6.7: Summary window of the new workspace.

The next step is the creation of the *mapset*, that is to say the subdirectory in which you will actually work. By default the PERMANENT mapset is proposed. It is advisable to create a new mapset by writing a name in the text box at the bottom of the window and clicking *Create*. The mapsets that will be loaded in the workspace are the ones that are checked (by checking the check box corresponding to the mapset name).

In *JGrass* it is possible to save and carry out operations on maps from different mapsets, however when you are working with native GRASS commands it is not possible to select the mapset and so reference is made to the default mapset, that is to say the first of the list of loaded mapsets. It is therefore advisable to use workspaces with only one loaded mapset, so as to be able to process data with the native GRASS commands[1], and to turn to multi-mapset mode only for transitory operations.

[1]By *native GRASS* we mean those commands that are executed in *JGrass* applying the original GRASS command. These operations are carried out unseen by the user as the commands have a standard *JGrass* type graphic interface.

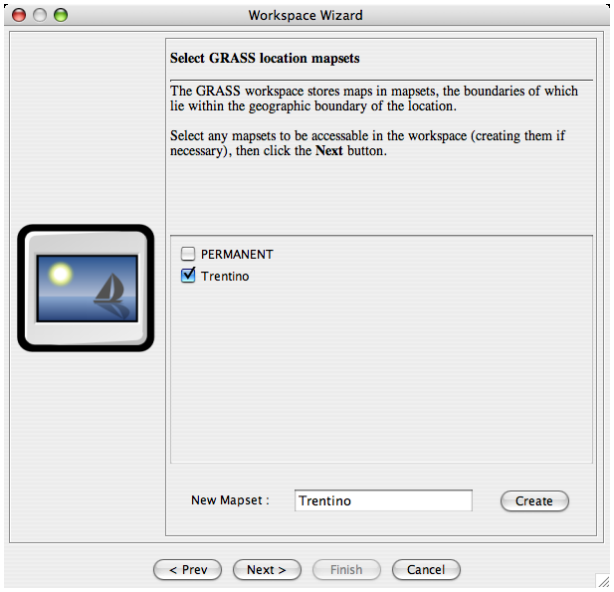

Figure 6.8: Window for the selection of the mapsets to load in the new workspace.

The next window asks for the name of the new workspace and a brief description, for example, you can write what is shown in figure 6.9.

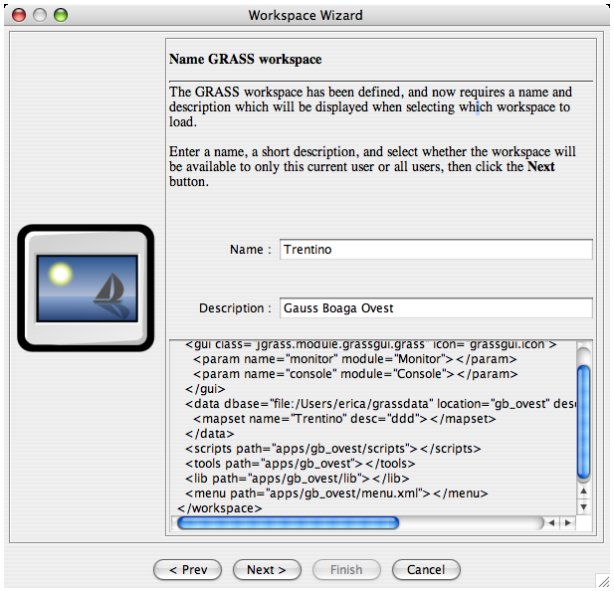

Figure 6.9: Window requiring the name to be displayed in the loading phase of the new workspace.

The last step of the wizard gives you the possibility to decide which toolbars to load in the new workspace by default, depending on the data processing that wish to carry out (see figure 6.10).

Click *Next* to finish the creation of the new workspace. If all went well the window shown in figure 6.11 appears. Click *Finish* to exit the wizard.

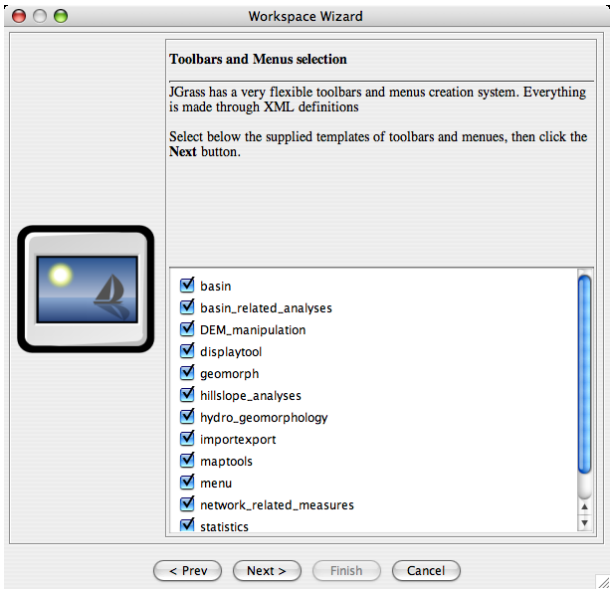

Figure 6.10: Dialog box for the selection of the toolbars of the new workspace.

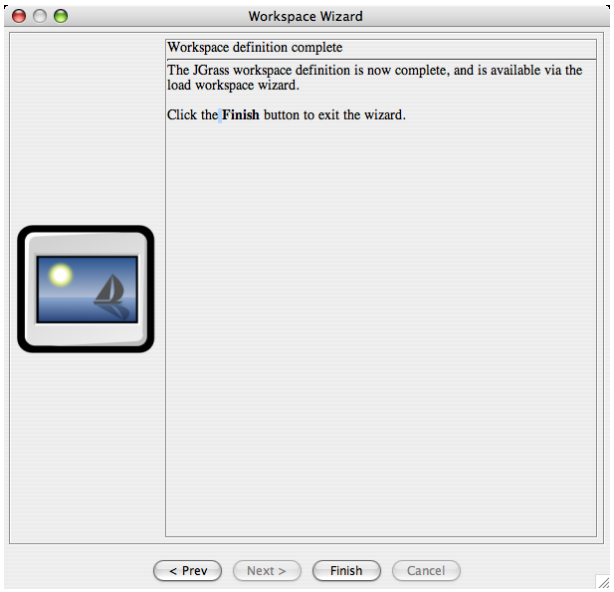
Workspace Wizard
Workspace definition complete
The JGrass workspace definition is now complete, and is available via the load workspace wizard.
Click the Finish button to exit the wizard.
< Prev
Next >
Finish
Cancel

Figure 6.11: Closing window of the workspace wizard for the creation of a new workspace.

The new workspace is now listed in the *load workspace* window (Figure 6.12) that appears by either clicking on the first icon of the toolbar or by selecting *Load Workspace* from the *File* menu.

6.1.1 Loading an Existing Location

To load a workspace that was created in GRASS or in a previous version of *JGrass* you need to follow the procedure described in the previous paragraph up to the point where you are asked

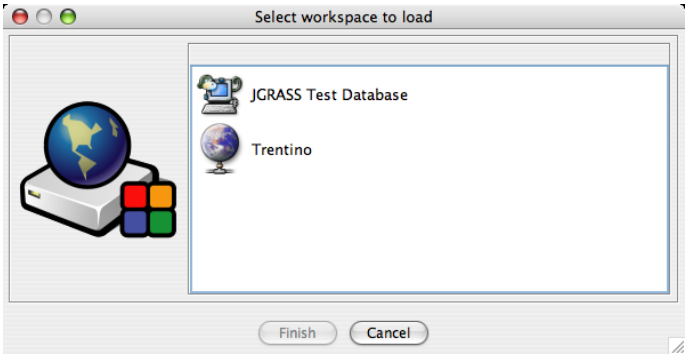

Figure 6.12: Window to select the workspace to load in *JGrass*.

to select the pathname for the creation of a new workspace (figure 6.4), but here select a path to an already existing location. At this point you will skip directly to the dialog box where you can select the mapset to work with (figure 6.8). Just as in the previous paragraph, here you can load one or more mapsets or create a new ones and load them to the workspace.

Even here, the new workspace will be listed with other available workspaces during the *load workspace* phase.

6.2 Choice of the Reference System and Cartographic Projection

The choice of the reference system and the cartographic projection to use is extremely important. We will now give some essential information which is, however, by no means exhaustive. Choosing the reference system means specifying a reference ellipsoid and its orientation, which define the planimetric reference surface of the portion of land of interest.

The cartographic projection defines the projection rules that allow us to pass from the representation on the curved reference surface (ellipsoid-Datum) to a plane reference surface (map).

Generally, for a given country, the cartography is produced using a particular reference system to which a particular cartographic projection is associated. In Italy, generally, the cartography is based on the Rome 1940 reference system and the Gauss-Boaga cartographic projection. More recently, along with other EU nations, the UTM projection and the ETRF89 reference system are being used. Often the term WGS84 is improperly used to refer to a reference system for data whose positions derive from post-processing of GPS measures. In GIS applications this inaccuracy does not have significant effects on the coordinates of the data, as long as the reference system used is global, such as the ITRF89, the WGS84 itself, and, for Europe, the ETRF89.

The European Petroleum Survey Group (EPSG) has standardized the codes that respect these types of association and so allow for an easier identification of the reference systems and projections used in different countries.

For coded digital maps in certain file formats it is possible, by means of a series of commands which shall be illustrated later, to create, on the basis of the file itself, a new location that is coherent with the map in terms of reference system and projection. This operation is only possible with those file formats that contain, other than the map, the meta-information that specifies the reference system and the projection. For those file formats that do not include this information (e.g. the ASCII formats) it is necessary to manually create a location which is coherent with the map that you want to import. In this case you can refer to EPSG codes that allow you to create a location without having to manually insert all the parameters that define the reference system and the cartographic projection of the map.

Specifically, in *JGrass*, in the window where you are asked to select the reference system and the projection (figure 6.6) it is possible to select the opportune EPSG code and so load the EPGS specified parameters automatically.

With reference to Italy, among the available options for the choice of reference system and projection the phrase *Gauss-Boaga on Rome 1940* does not appear. This combination is indicated internationally with two titles, depending on the longitude associated to the Italian reference meridian, the one that passes through *Rome - Monte Mario*.

- `Italy zone 1` and `Italy zone 2` (depending on the zone to use) if the longitude of the reference meridian is referred to the Greenwich meridian. In this case, the longitude of the reference meridian is: 12 degrees 27 minutes 8.4 seconds. The longitude of the central meridian of zone West (or zone 1) is: 9 degrees, while for zone East (or zone 2) it is 15 degrees.

- `Monte Mario / Italy zone 1` and `Monte Mario / Italy zone 2` (depending on the zone to use) if the longitude of the reference meridian is set to 0 degrees. The longitude of the central meridian of zone West (or zone 1) is: -3 degrees 27 minutes 8.399 seconds, while for zone East (or zone 2) it is 2 degrees 32 minutes 51.6 seconds.

The precise wording and the corresponding codes for the Gauss Boaga on Rome 1940 cartographic projection and reference system used in Italy are now listed:

3003: **Monte Mario / Italy zone 1**
3004: **Monte Mario / Italy zone 2**
26591: **Monte Mario (Rome) / Italy zone 1**

26592: **Monte Mario (Rome) / Italy zone 2**

Should it be necessary to work with geographic coordinates (ϕ and λ), referring to the reference ellipsoid, rather than cartographic coordinates (N and E), it is still possible to specify only the reference ellipsoid and its orientation (Datum). In this case the EPSG code to create a location with geographic coordinates referred to the Rome 1940 reference system is:

4806: **Monte Mario (Rome)**

Should it be necessary to create a location in which to import data acquired with commercial GPS instruments, that are therefore not post-processed, or data that comes from official cartography the EPSG reference codes are:

- for geographic coordinates

  4326: **WGS 84 / latlong**

- for UTM cartographic coordinates (Italian zones)

  32632: **WGS 84 / UTM zone 32N**
  32633: **WGS 84 / UTM zone 33N**

In the case of post-processed and ETRF89 compliant GPS data:

- for geographic coordinates

  4258: **ETRS89**

- for UTM cartographic coordinates (Italian zones)

  25832: **ETRS89 / UTM zone 32N**
  25833: **ETRS89 / UTM zone 33N**

In the case of coordinates in the ED50 (EuropeanDatum 1950) reference system:

- for geographic coordinates

  4230: **ED50**

- for UTM cartographic coordinates (Italian zones)

  23032: **ED50 / UTM zone 32N**
  23033: **ED50 / UTM zone 33N**

An example of UTM ED50 cartography are the “excursion” type maps printed by Tabacco, and Compass.

NOTA BENE

1. For all GIS applications, the difference in coordinates between the WGS84 system and its ETRF89 materialization is not appreciable and therefore negligible. The creation of a location in either one of the two systems is equivalent.
2. There is no significant difference, in this context, between writing ETRF89 and ETRS89.
3. The difference between coordinates expressed in the ED50 system and the ETRF89 or WGS84 system is , on average, equal to about 200 metres for the entire Italian territory.

6.3 Importing Raster Data

Once you have created a workspace with the right reference system it is indispensable to import the data that you wish to process. You can start either from raster maps or vector maps.

6.3.1 Importing Raster Maps

To import a raster map there are different possibilities, according to the format of the map that you want to import to *JGrass*:

1. use the r.in.ascii command
2. use the r.in.lidar command
3. use the r.in.gdal command
4. copy the file to one of the *JGrass* directories[2].

The r.in.ascii Command

With the *r.in.ascii* command it is possible to import ASCII files in the characteristic GRASS format or in the ESRI GRID format.

The graphic interface for the command is shown in figure 6.13, it contains:

[2]It is sufficient to copy the file, from outside of *JGrass*, to the mapset directory with the name corresponding to the file format, if this is supported. In this way the map becomes accessible for processing

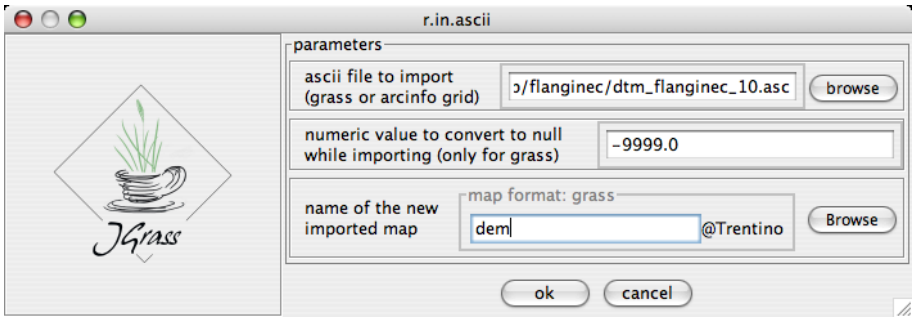

Figure 6.13: The *r.in.ascii* graphic interface in *JGrass*.

- the pathname for the file to be imported, you can select it by clicking *browse*;
- the value in the map that represents a NOVALUE (to know it you will need to open the file with any text editor and, normally, the data value that represents a NOVALUE is indicated);
- the name of the imported map (you can even select the destination mapset if you wish).

The map created in the example is *dem* and it can be viewed as shown in figure 6.14. In this case, the working resolution is 10m, which is exactly the same as that of the imported map, but the active data region is decidedly bigger than the map. Working in this way, with an active data region which is bigger than the map that is being processed, causes waste of time and memory resources, it is therefore advisable to redefine the region parameters.

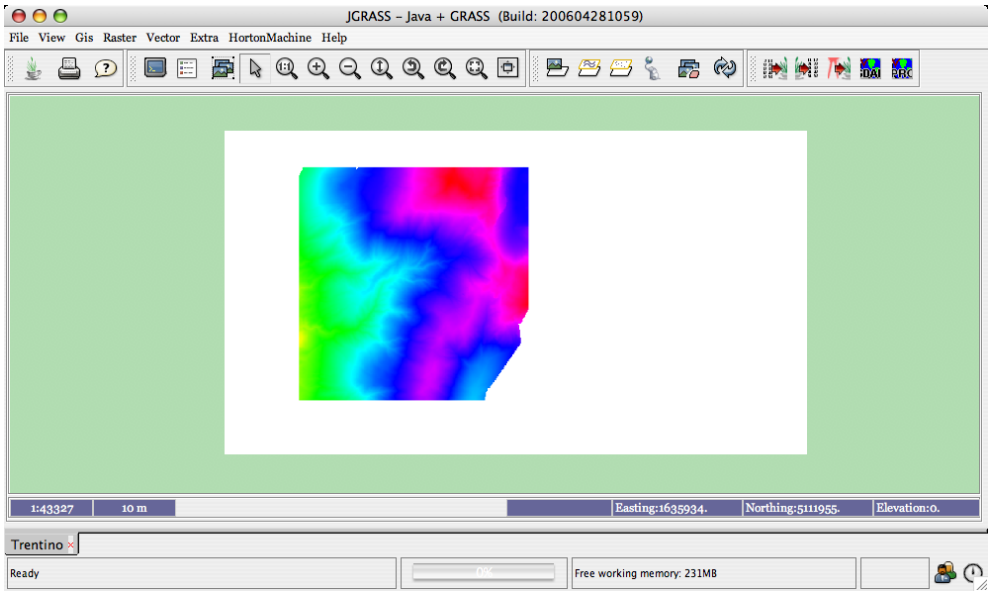

Figure 6.14: Visualization of a map imported with the *r.in.ascii* command.

To define the active data region parameters you can use the two methods described in Paragraph 5.2. In the first example, in figure 6.15, all the information to set the active data region to the same dimensions as the imported map are given. It is sufficient to select the map in the *map selection* text box and then click the *set region from map* button. You will realise that by carrying out this operation the region limits are set automatically.

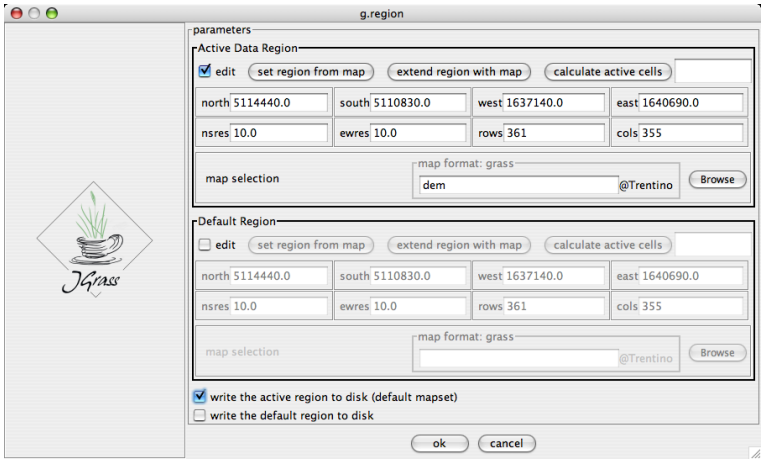

Figure 6.15: Graphic interface of the region manager tool used to set the active data region to the dimensions of the imported map.

A view of the map with the new region settings is shown in figure 6.16. Take note that for a better view the zoom parameters were adjusted and the map was redrawn to the current display resolution by clicking on the redraw icon in the map display toolbar.

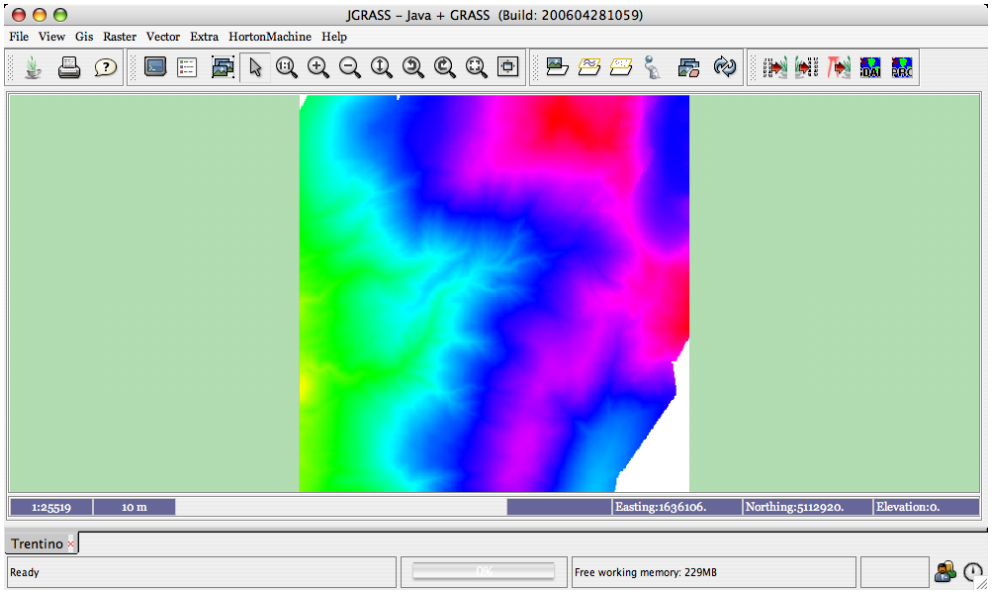

Figure 6.16: Visualization of the map imported with the *r.in.ascii* command.

In this way it is possible to import all the GRASS and ESRI ASCII raster maps you need for future processing.

The r.in.lidar Command

By means of the *r.in.lidar* command you can import data saved in point triplets format corresponding to the x,y,z coordinates to *JGrass*. The name of the command comes from the fact the data from altimetric laser surveys are often given in this format. Of course you can also import data from other software applications or from other types of surveys as long as you maintain this format. The graphic interface for the command is shown in figure 6.17. The information it asks for is:

1. the name of the file to be imported;
2. the specification whether the data are on a regular data grid or irregularly dispersed over an area;
3. the name of the new map
4. the specification whether to import only the data corresponding to the active data region or all the data on the file (operation that in most cases requires a lot of time)
5. the specification of what to do with duplicated points, whether to take the maximum, the minimum, or a mean of the values relative to a single point (this is to decide whether to

consider the vegetation or the ground, if necessary). This option is not considered if there are no duplicated points in the map.

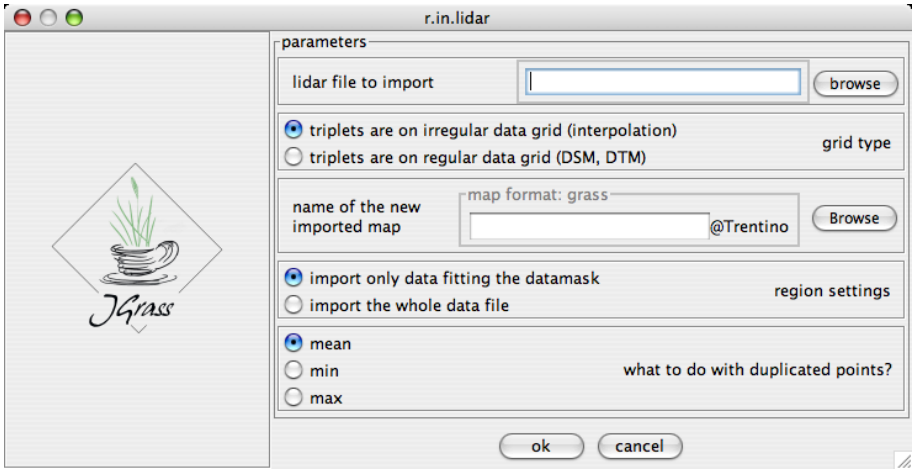

Figure 6.17: The *r.in.lidar* graphic interface in *JGrass*.

The r.in.gdal Command

This command is one of the native GRASS commands. It allows you to import all the formats that are supported by the GDAL libraries[3]. The graphic interface is very simple, it allows you to select the input file in any directory by means of the *browse* button and to define the operations that can be carried out it:

1. select the colour band to import when importing image files, if no colour is specified all the colour bands are imported and a raster is created for each band (a suffix relative to the colour is added to the name, e.g. *.red*);
2. define the location name where the data projection is to be read;
3. define the name of the new location to be created if the projection characteristics should not be congruent with those of the current workspace;
4. ignore the information about the region limits so as to be able to import even maps that are outside the active data region;
5. extend the region limits to those of the imported map;
6. list the supported formats;

[3] www.gdal.org

7. consider the band number rather than the name of the colour when importing images.

In the example given, the option that allows you to import maps outside the active data region has been selected, the map to be imported is a technical map of an area near the DEM area but does not coincide with it.

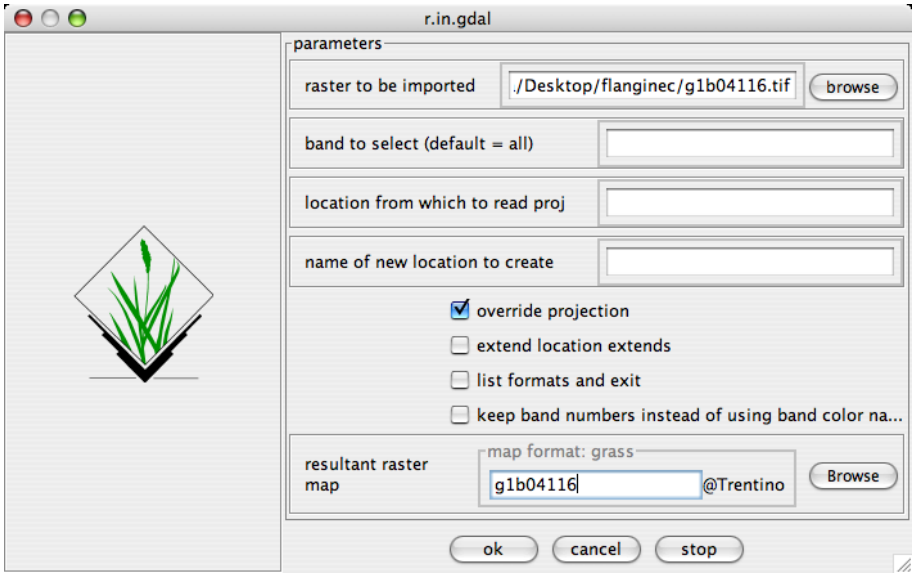

Figure 6.18: The *r.in.gdal* graphic interface in *JGrass*.

Once the parameters to be used during the processing phase have been defined all you have to do is click *ok* to carry out the operation. The operation progress is displayed in the graphic interface as an output of the original GRASS command. If you realise that there are incorrect data or for some reason you have to interrupt the program, all you have to do is click the *stop* button to terminate the operation before obtaining the final result.

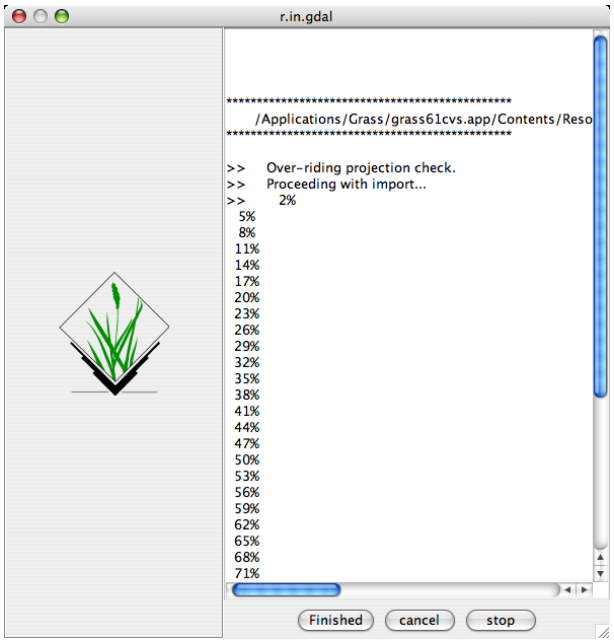

Figure 6.19: Output of the *JGrass r.in.gdal* command.

If you now try to display the imported map a message will appear warning you that the map is completely external to the displayed region. You must redefine the region parameters or extend the region to include the new map (figure 6.20), or you can set the region limits to the newly imported map (figure 6.21). In this second case it will also be necessary to edit the default region and set it to the same map or extend it, otherwise you would have the inconsistency of an active data region outside the default region.

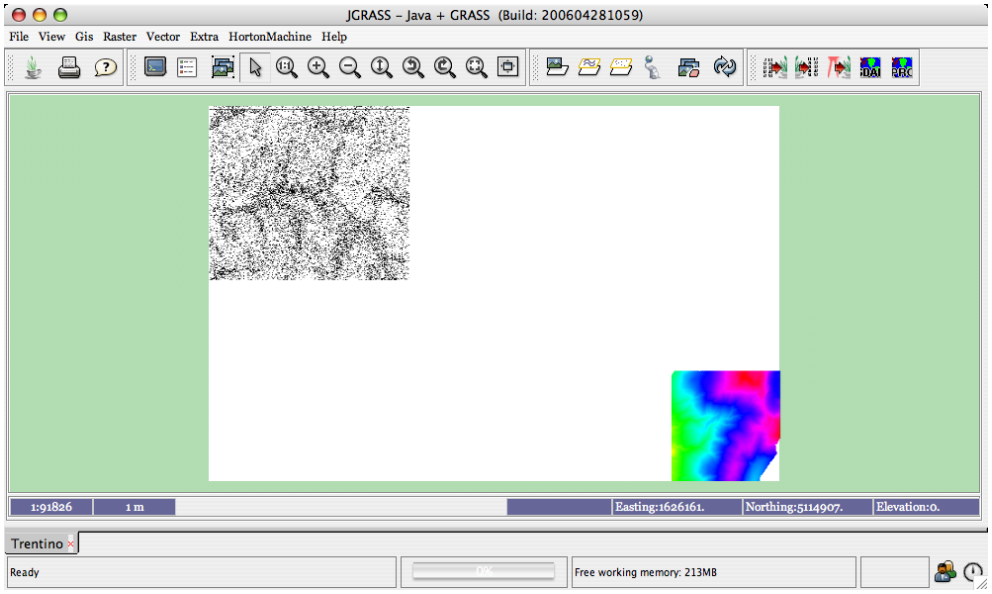

Figure 6.20: Extension of the active data region to include the new map imported with the *r.in.gdal* command.

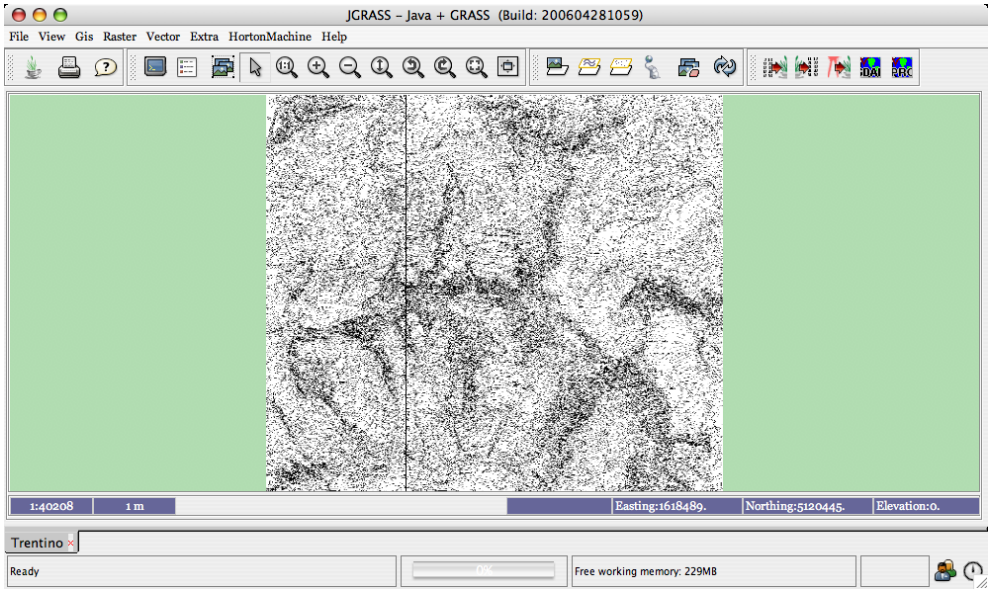

Figure 6.21: Setting the active data region to that of the new map imported with the *r.in.gdal* command.

In the case shown, if you zoom-in, the details of the provincial technical map of Trentino can be seen very well (figure 6.22). The resolution of the new workspace, if the region is set to the new imported map, passes from 10 metres to 1 metre. The map scale and the working resolution are displayed in the lower part of the screen. If you use such extreme zooms, by using a window zoom on a part of the map, it is better to redraw the map at the current zoom in order to view it adequately.

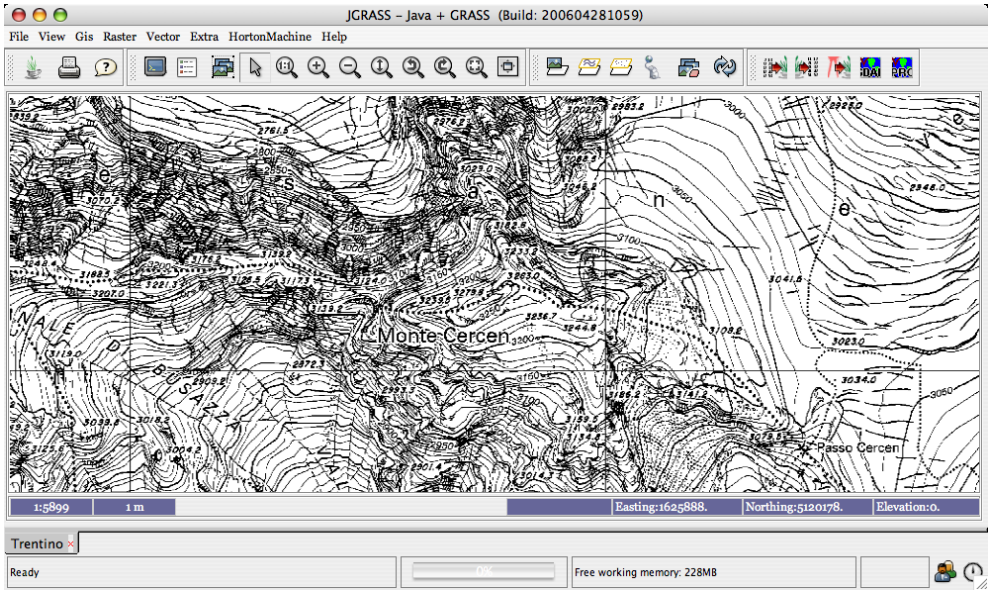

Figure 6.22: Enlargement of a portion of the technical map of the Province of Trento, imported with the *r.in.gdal* command.

6.3.2 Importing Vector Maps

To import vector maps it is sufficient to view them with the *d.shapefile* command (Paragraph) and select the directory of origin. In *JGrass* the shapefile format is kept as a swap format and all processing starts off from this format. If you use the native GRASS commands, that have their own specific format for vector maps, the command itself will carry out all further processing to pass from one format to another, you only have to specify the names of initial and final maps, which will be in shapefile format.

6.4 Data Re-Projection

In this section, the *r.proj* command will be explained, which allows you to re-project raster maps from one reference system to another. This command is also part of the set of native GRASS commands implemented in *JGrass*. The version for data in vector format also exists, but as of yet the graphic interface has not been created. All the native GRASS command for which a graphic interface has not yet been created are executable from the console (Chapter 9).

As far as raster maps are concerned, to re-project a map from one location to another characterized by a different cartographic projection you must do as follows. Should it be necessary to pass data from the Gauss Boaga zone West projection to the UTM-WGS84 projection, two operations are required:

1. the creation of a location with the desired projection (in this case UTM-WGS84)

2. re-project the necessary data with the *r.proj* command

For the first operation you only have to repeat the steps described in Paragraph 6.1: *Creation of a Location*, but changing the projection parameters to the ones corresponding to the UTM-WGS84 projection (Paragraph 6.2). The process is exactly the same as the one described earlier except for the definition of the projection parameters.

The projection parameters to input in the window shown in figure 6.6 for the creation of a location with a UTM-WGS84 projection are given below. It must be remembered that Italy is partly in zone 32 and partly in zone 33 of the northern hemisphere in the UTM projection representation. You will have to select from the EPSG file, by means of the graphic interface, the parameters corresponding to:

32632: **WGS 84 / UTM zone 32N**

By writing UTM WGS84 in the provided text box and scrolling through the results in the dropdown menu you will be able to identify the desired projection as shown in figure 6.23.

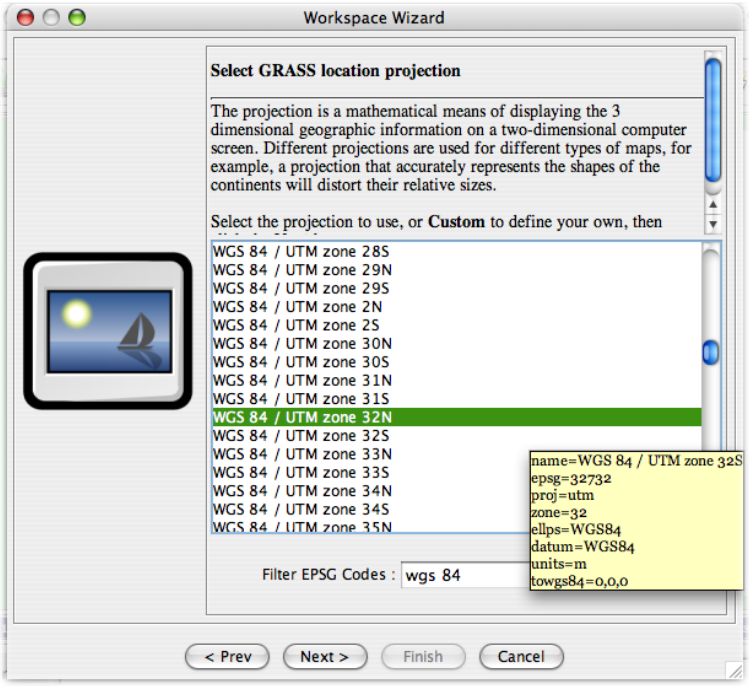
Workspace Wizard
Select GRASS location projection
The projection is a mathematical means of displaying the 3 dimensional geographic information on a two-dimensional computer screen. Different projections are used for different types of maps, for example, a projection that accurately represents the shapes of the continents will distort their relative sizes.
Select the projection to use, or Custom to define your own, then
WGS 84 / UTM zone 28S
WGS 84 / UTM zone 29N
WGS 84 / UTM zone 29S
WGS 84 / UTM zone 2N
WGS 84 / UTM zone 2S
WGS 84 / UTM zone 30N
WGS 84 / UTM zone 30S
WGS 84 / UTM zone 31N
WGS 84 / UTM zone 31S
WGS 84 / UTM zone 32N
WGS 84 / UTM zone 32S
WGS 84 / UTM zone 33N
WGS 84 / UTM zone 33S
WGS 84 / UTM zone 34N
WGS 84 / UTM zone 34S
WGS 84 / UTM zone 35N
name=WGS 84 / UTM zone 32S
epsg=32732
proj=utm
zone=32
ellps=WGS84
datum=WGS84
units=m
towgs84=0,0,0
Filter EPSG Codes : wgs 84
< Prev
Next >
Finish
Cancel

Figure 6.23: Choosing the UTM-WGS84 projection parameters in the workspace wizard for the creation of a new workspace.

For better data management, values are given to the region limits, even in this case the values are typical for Trentino. The result is shown in figure 6.24. Clicking *Next*to continue to the successive

steps that allow you to define the working mapset and the toolbars to be loaded.

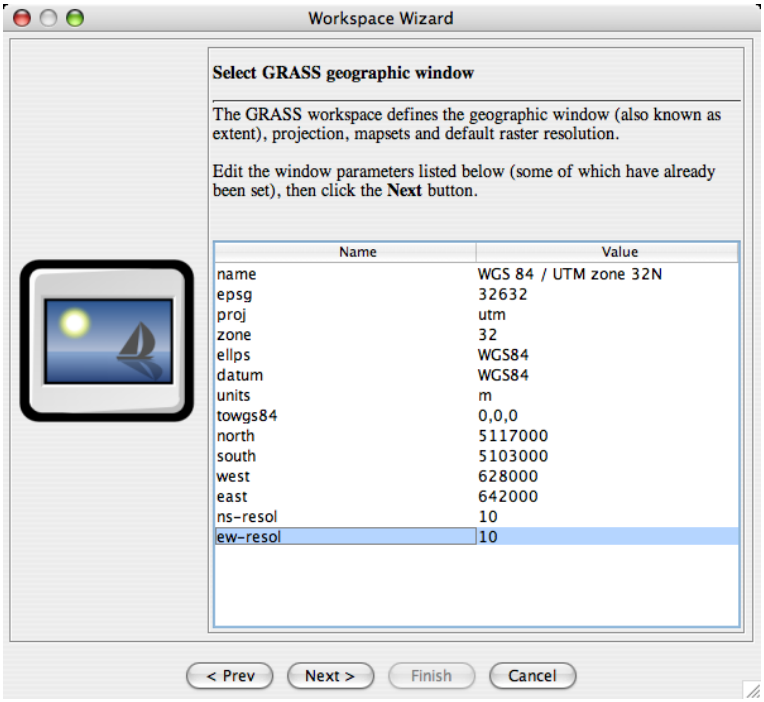

Figure 6.24: Summary window of the UTM-WGS84 projection parameters in the workspace wizard for the creation of a new workspace.

The name chosen to be displayed in the initial window of the *load workspace* wizard, when you have data in different projections, should contain some indication of its projection, in this case, for example, you can call it *trentino utm*.

Once the new workspace with UTM projection has been created, you can access it and, with the *r.proj* command, re-project the data from the Gauss Boaga zone West location. This command requires as input:

- the name of the mapset with the input map, which is selected by means of the *browse* button
- the name of the input map, which must be inserted manually
- the name of the output map
- the resolution of the output map, which is already set to the resolution value of the current workspace
- the interpolation method to be used
- the choice to list the maps present in the input location or not
- the choice to optimize the cutting of the map to the limits of the active data region or not

The tool presents itself as shown in figure 6.25, where the *dem* map is being re-projected to a new reference system.

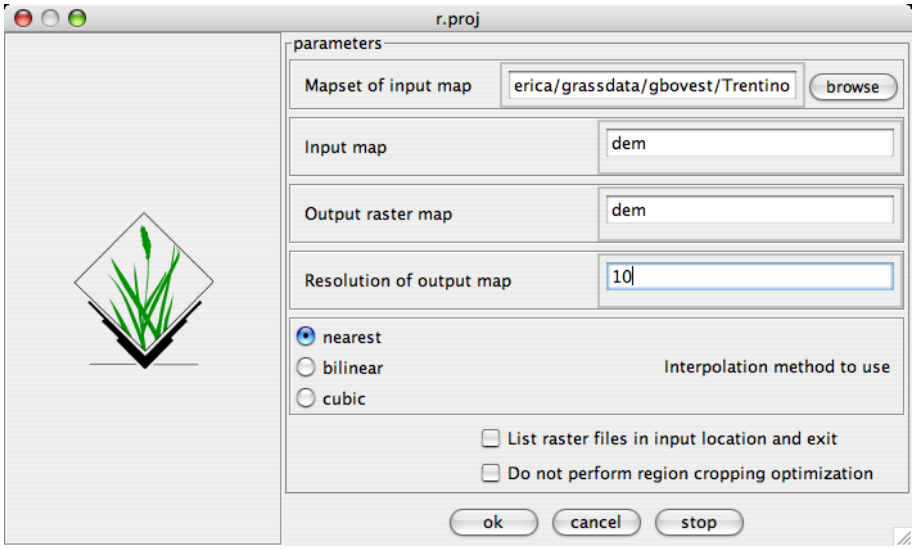

Figure 6.25: The *r.proj* graphic interface in *JGrass*.

Once the operation is complete it will be possible to view the *dem* map and continue with other operations in both the Gauss Boaga zone West location and the UTM-WGS84 location indifferently.

7 Special Operations

In this paragraph some useful solutions for raster and vector map processing will be presented. These solutions have been prepackaged for the user, but they are not necessarily the only path to the required solution.

7.1 Creating Contour Lines from a DEM

To carry out a geomorphological analysis a DEM in raster format is required. However, it is often necessary to export the data for graphic processing with other applications, it is therefore useful to have a contour lines file created from the DEM data. The tool to use in this case is *r.dem2contour*, its graphic interface is shown in figure 7.1.

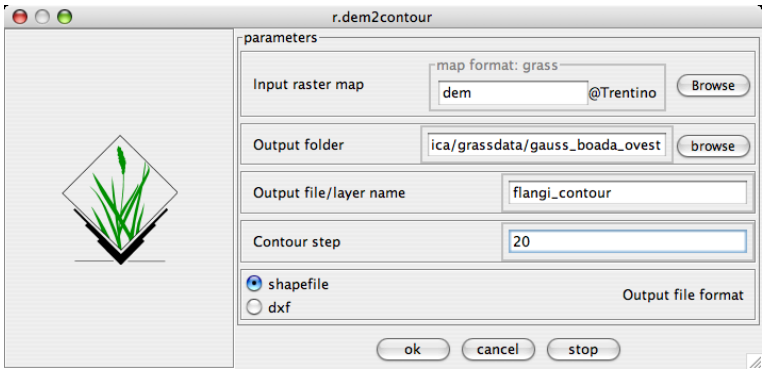

Figure 7.1: The *r.dem2contour* graphic interface in *JGrass*.

The parameters to be inserted are:

1. name of the input raster map from which to create the contour lines,
2. name of the destination directory where the output file in shapefile format is to be saved,
3. name of the output file
4. the contour line step[1],
5. output file format: shapefile or dxf.

Even here, the command processing is carried out within the graphic interface, you can interrupt the program at any time or, at the end of the operation, scroll through the output messages.

[1]The contour line step must be greater than the resolution of the input map, otherwise you risk forcing contour lines on non-real data.

The fact that, in some cases, there are error messages during the operation does not necessarily compromise the successful result of the program. Viewing the output file will result in something similar to figure 7.2.

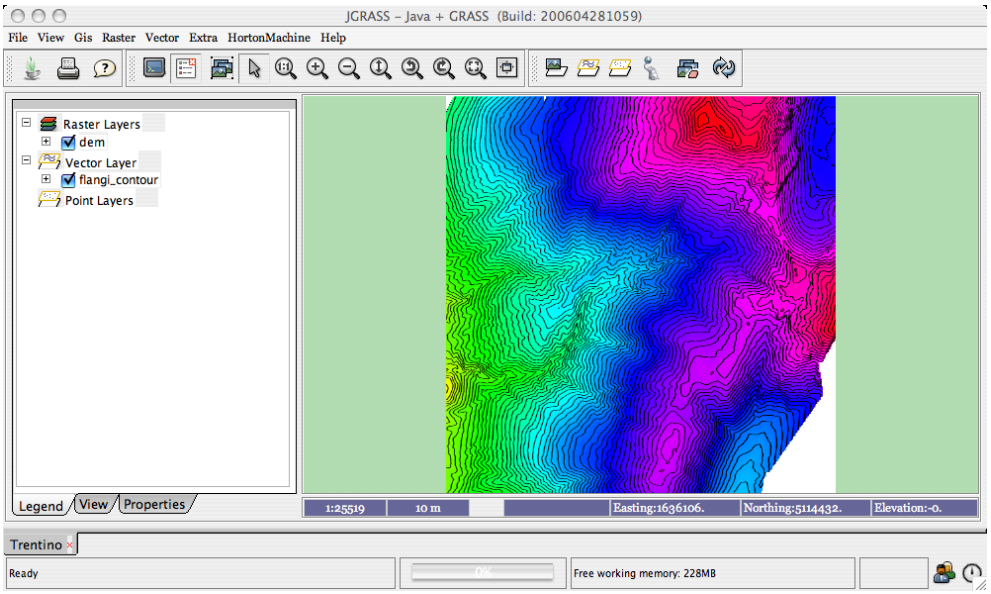

Figure 7.2: The resulting map from the *r.dem2contour* operation superimposed onto the original terrain model.

7.2 Creating a DEM from Contour Lines

Just as useful is the possibility to create a DEM from contour line data. If the contour data is in shapefile format it is possible to read the file and interpolate the contour lines to obtain a map of the terrain model. The command to use in this case is *v.contour2dem* from the *Vector* menu (figure 7.3). This command is very simple and requires, as input data, the name of the shapefile with the contour lines, the destination folder for the output file, and the name of the output file. The resolution of the output map will be that of the current mapset, the one displayed in the lower part of the screen.

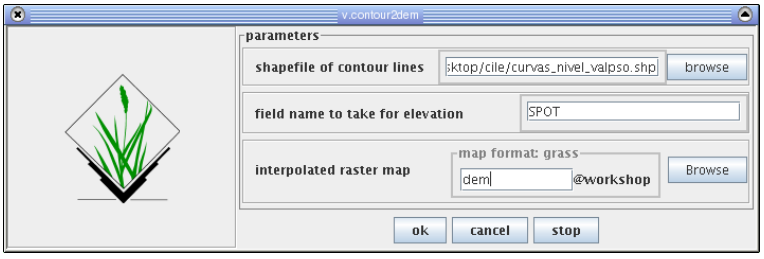

Figure 7.3: The *v.contour2dem* graphic interface in *JGrass*.

The process of interpolating the contour lines to obtain a raster requires a lot of time, by all means the operation's progress is displayed in the command window. Here it is also possible to check how the process is going and if there are any errors. These cannot not be modified during the operation, if anything, the operation can be interrupted, the error fixed and the operation repeated from the beginning.

When using native GRASS commands it is of utmost importance that the output maps do not exist already. If a map exists with the same name as the GRASS process output map the operation will be terminated with an error of the "map already exists" type. Particularly in the case of the *v.contour2dem*, various GRASS commands have been incorporated, this is why the operation output is such a wealth of information. The result of the interpolation is shown in figure 7.4, the original contour lines are also shown in transparency. Of course, calculations can be carried out on the resulting map to eliminate any inconsistencies or to isolate a part of the region. For example, the yellow area to the North of the map in figure 7.4 could be removed with the r.mapcalc tool by assigning *null* to the value corresponding to the yellow of the map.

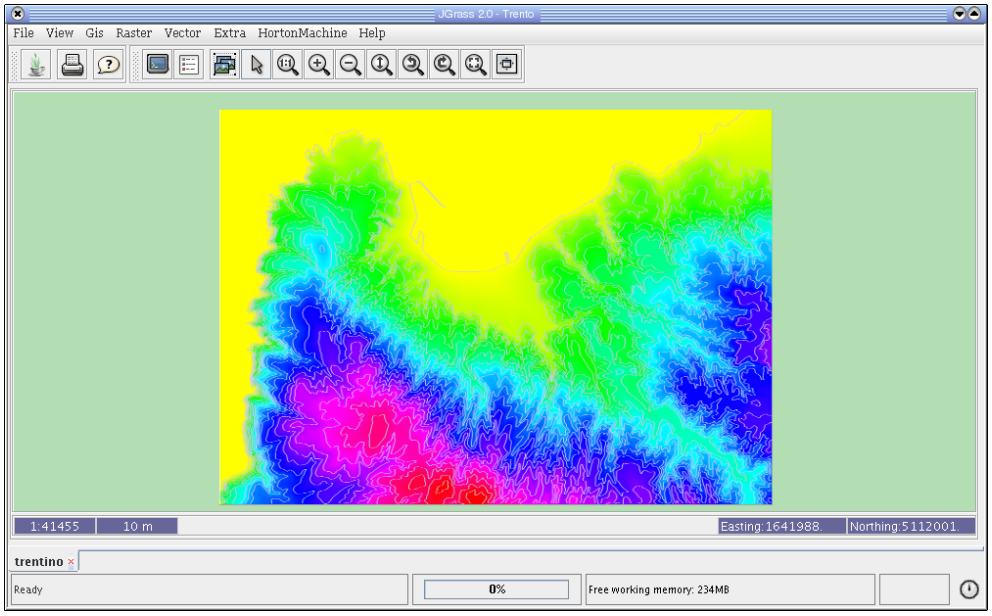

Figure 7.4: Result of the interpolation done with the *JGrass v.contour2dem* command.

8 Environmental Databases

Most of the environmental monitoring information made available by designated institutions has spatial information associated to it. This is why GIS are used as containers and viewers for all sorts of environmental data.

If the amount of available data is greater than a critical limit for the good-functioning of a PC or if, by all means, the data have to be shared among various users, then databases are used. There are different categories of databases. In the environmental field we usually refer to two types of database, point databases and spatial databases. The distinction between the point database and the spatial database is precisely in their contents.

Point databases contain information taken in known points and therefore they contain tables with the coordinates of the points, the altitude, and the measures taken, inserted as attributes in the table (all these data are fields in the database tables).

Spatial databases, on the other hand, have a field reserved to the geography and geometry of the inserted object which refers to a specific table that contains its geometric description. In these data the projection and datum are also considered, as well as the relationship between objects (topology).

For a database connection in *JGrass* it is necessary to define the name and position of the database, the connection will be through a database client.

8.1 Installation and Usage of a Database Client

The database client used here is called *SQuirreL SQL*[1], this too is distributed with an Open Source licence, all you need to do is download the software from the web site and install it. The downloaded file is a *.jar* file and therefore the installation and the problems that may arise under different operating systems are similar to those presented in Chapter 3: Installation[2].

SQuirreL SQL Client is a Java-based program with graphic interface which allows you to see the structure and consult the tables and data of a database through JDBC libraries.

The client can be interfaced with different types of databases by simply loading the drivers that can be downloaded directly from the web site. In the following example the connection to the PostgreSQL Open Source database is shown. Once the basic software has been installed all you

[1] http://squirrel-sql.sourceforge.net/

[2] Some practical notes for the installation of this particular software are found at: http://squirrel-sql.sourceforge.net/

have to do is copy the *jar* file corresponding to the desired extension to the *lib* directory. For the PostgreSQL database the driver to install is called *postgresql-8.0-314.jdbc3.jar*.

8.1.1 Creation of the Remote Database Connections

To define the characteristics of the database connection you need to launch SQuirreL and create a new "alias" connection. The first window of the SQuirreL start-up is shown in figure 8.1.

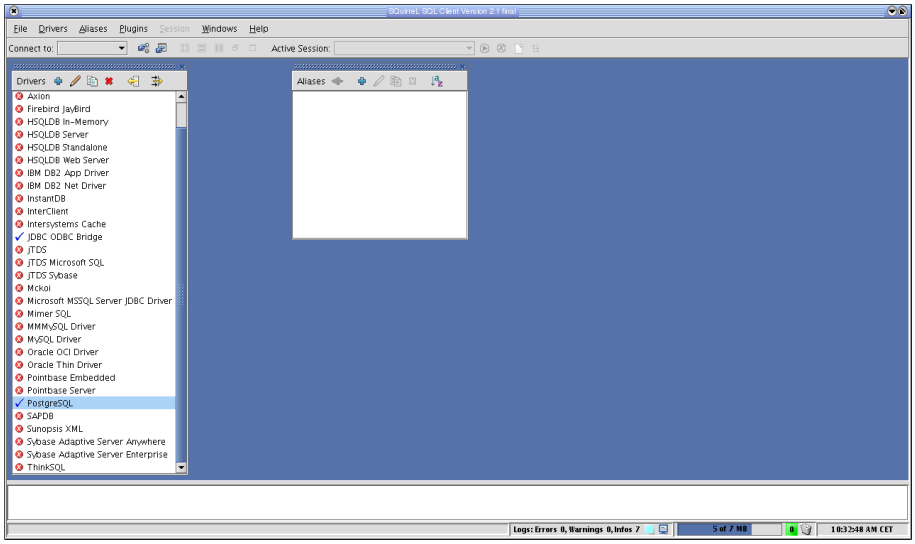

Figure 8.1: The SQuirreL start-up screen.

The important things to notice are the two windows displayed by the program that contain the list of available and installed drivers, on the left, and a list of the databases to which you can connect, in the centre. In the example, it is important to check that the PostgreSQL driver has been loaded. At this point you can create a new database connection by clicking on the + of the *aliases* table (Figure 8.2). The required information when creating a new connection is:

1. name of the connection
2. the driver used (the type of database to which to connect)
3. the URL of the database, written with the specific syntax that is suggested at the beginning of the procedure
4. username
5. password

All the other required information can be left with its default setting.

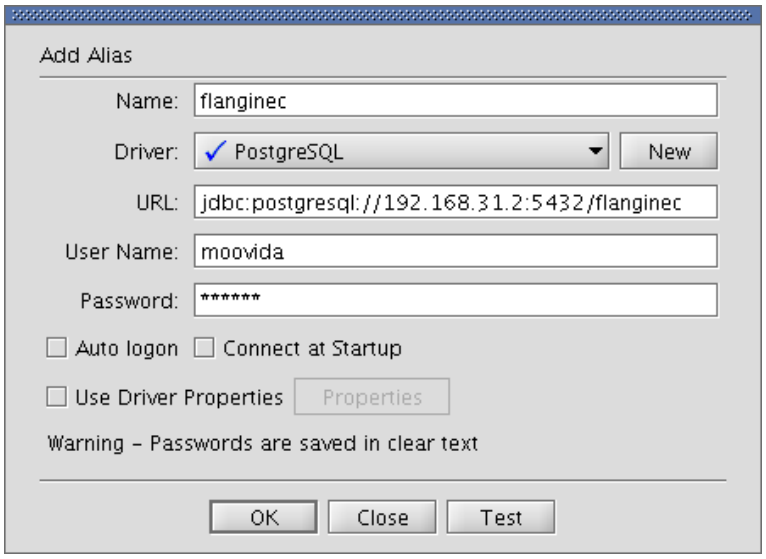

Figure 8.2: Creation of a new database connection in SQuirreL.

In this way you can connect to point databases, such as flanginec (Trentino), and spatial databases indifferently by specifying the connection name and the source URL. For example, you can connect to the spatial database of Osnabrueck (figure 8.3) or to the spearfish database (figure 8.4).

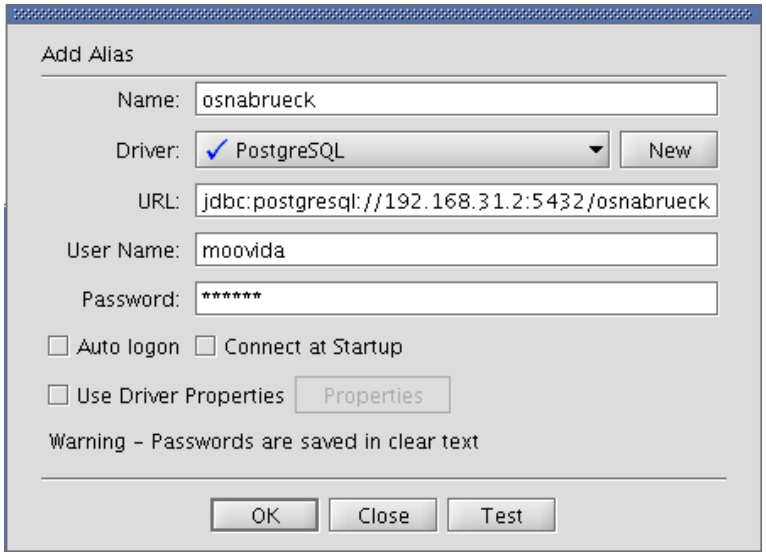

Figure 8.3: Creation of a new spatial database connection in SQuirreL.

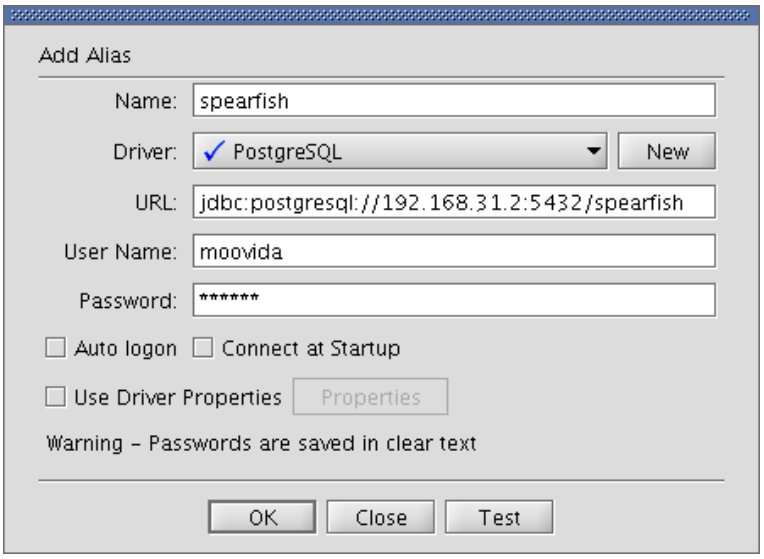

Figure 8.4: Creation of a new spatial database connection in SQuirreL.

8.1.2 Connection and Data Visualization

To access a database from *JGrass* first and foremost you need to establish a connection with SQirreL. This connection must remain active for the duration of the visualization and use of the data from the database.

To connect to a created database you just have to double-click on the name of the database or select the option from the dropdown menu. Once all the connection data have been confirmed it is possible to view the database structure. If, at the first access, the complete structure does not appear (see figure 8.5) just select *refresh* from the popup menu that appears by right-clicking on the database name.

It is not possible, at this time, to look into the database structure and all the connected information with great detail, despite the fact that it is a very simple database insomuch that it is a point database. The only information that is strictly necessary in order to understand the data is that which is found in the *public - TABLE* directory. Under each one of the tabs of the tables present (in the example there is only the *datizinco* table) you can find different types of information about the data. The most important information is under the *Columns* and *Content* tabs. Data referring to zinc concentrations measured at the ground and found in the *datizinco* table are shown in figures 8.6 and 8.7.

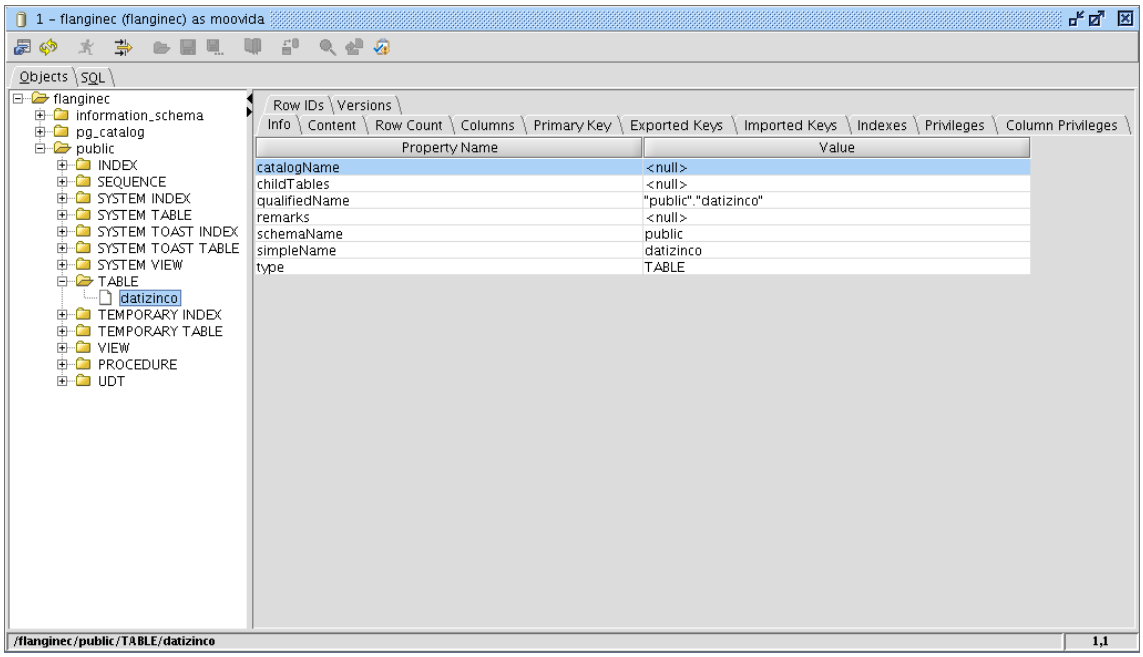

Figure 8.5: The flanginec database structure in SQuirreL.

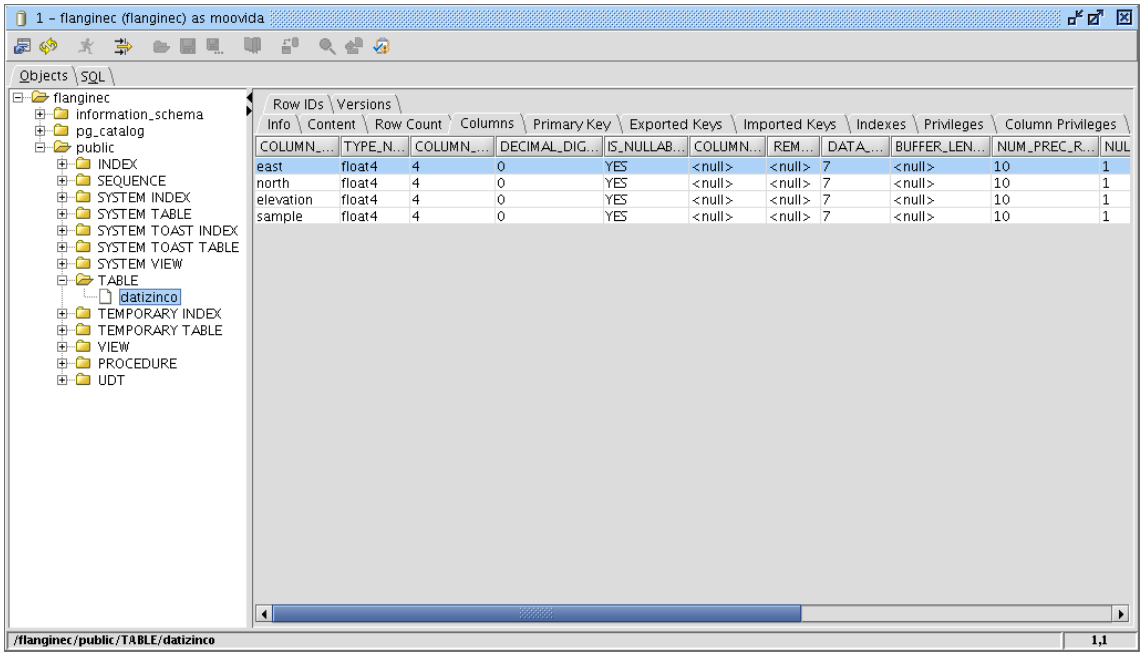

Figure 8.6: Type of data in the flanginec database (*Columns* tab) viewed with SQuirreL.

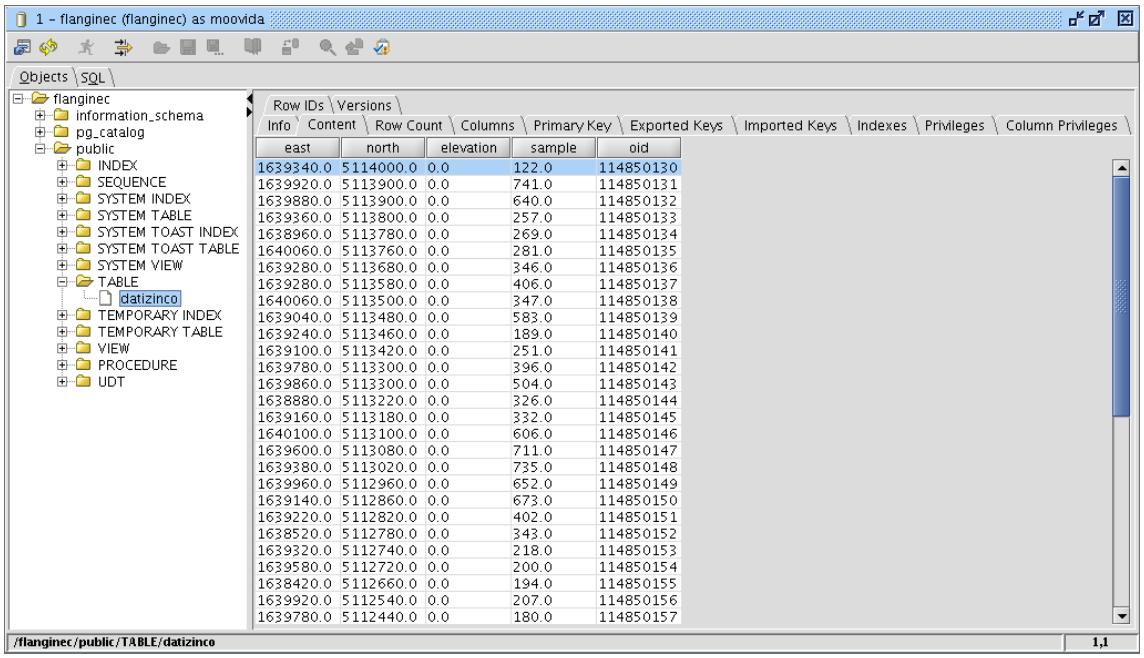

Figure 8.7: Viewing data from the flanginec database (*Content* tab) in SQuirreL.

At this point you have all the information necessary to process the data in *JGrass*.

8.2 How to Interface JGrass to the Database

The connection created in SQuirreL is not enough for *JGrass* to access the data. It is a rather the starting point, and surely the necessary condition, in order to to use the data. To access the data you need to give some information to *JGrass* regarding the type of database to connect to, and this information varies depending on whether you are dealing with a point database or a spatial database.

8.2.1 The Interface for a Point Database

Point databases, not having geometric information, have to be integrated into a *JGrass* workspace. The information about the point databases is in a file in the *sql* directory of the *mapset* (Paragraph 3.5). If this directory was not made during the creation of the new workspace, all you have to do is create it externally to *JGrass* with the classic commands for creating directories in a filesystem.

There are no graphic modes to create the file, you will have write the file, according to the specifications of the given example, with a text editor and save it with a *.sql* extension in the *sql* directory of the *mapset*. The extension is of the utmost importance insomuch as *JGrass* uses this

type of file for the database connection.

The information in the connection file is substantially the same as that used for the connection from SQuirreL, with the typical syntax of the xml language. The file must have:

- the name of the dataset
  - the type of driver used for the JDBC connection
  - the database URL (the same as the one used in SQirreL)
  - username
  - password
- the information (fields) to read
  - the column name
  - the data type stored in the column
- the data query to extract data from the database

An example of a *sql* file to connect to the flanginec database is shown in figure 8.8, obviously the specifications can vary according to specific circumstances, in particular, if you interface with the flanginec database the data will remain the same but you will have to specify the correct URL for the database.

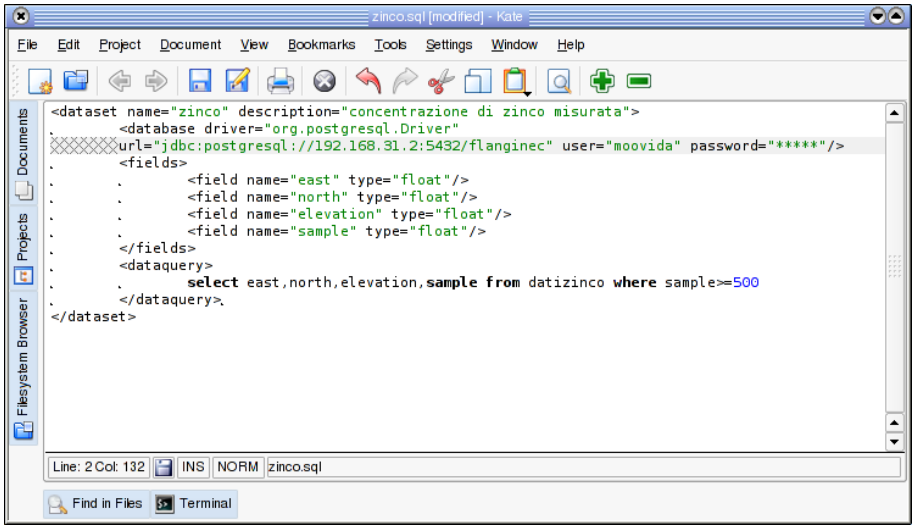

Figure 8.8: Example of an sql file to connect to a point database in *JGrass*.

As shown in figure 8.8, the syntax with which the data is inserted is extremely important. Take note that each field of the file is opened with the <> symbols and closed with < / >.

Database Querying

The "questions" that are sent to a database in order to extract information are called *queries*. To write queries you must respect specific parameters and syntaxes. In SQuirreL you can carry out queries directly with the graphic editor. In this way you can check the query results and, if necessary, modify the query in order to obtain the parameters needed for the following processing operations. Figures 8.9 and 8.10 show examples of two queries. In the first example, all the fields of the *datizinco* table are extracted, while in the second example only the north and east field of the same table are extracted.

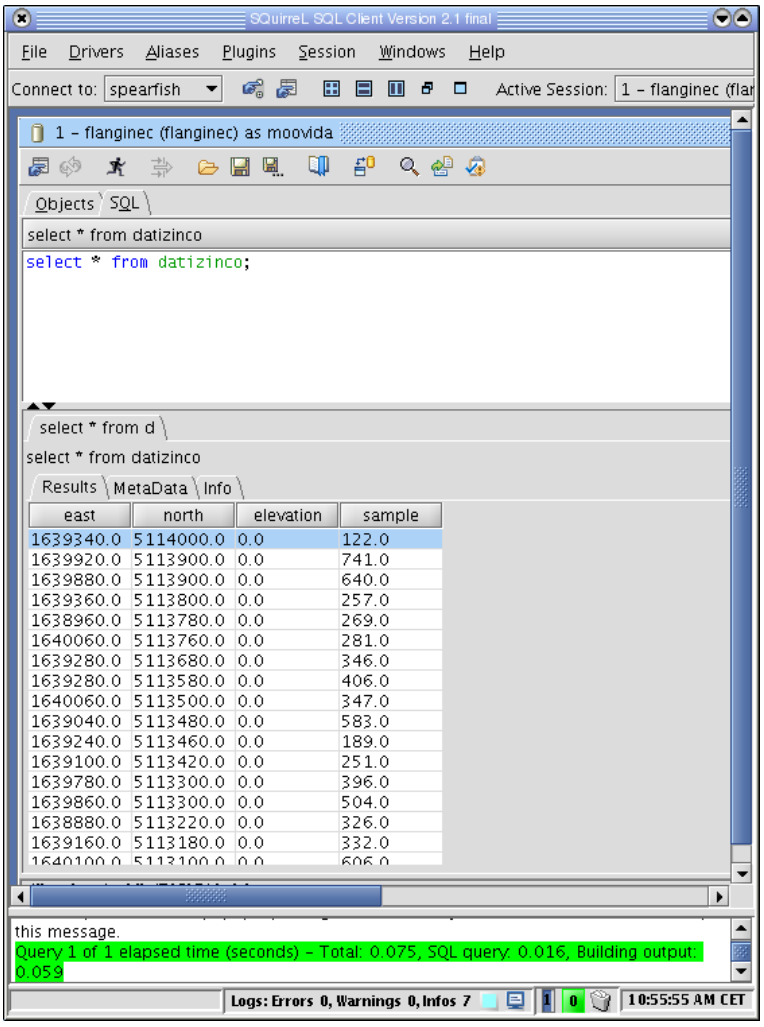

Figure 8.9: Example of a point database query in SQuirreL.

The query that was written to file in *JGrass* is the one shown in figure 8.11. It extracts data from the east, north, elevation, and sample fields for points with zinc concentration values at the ground (sample field) greater than 500.

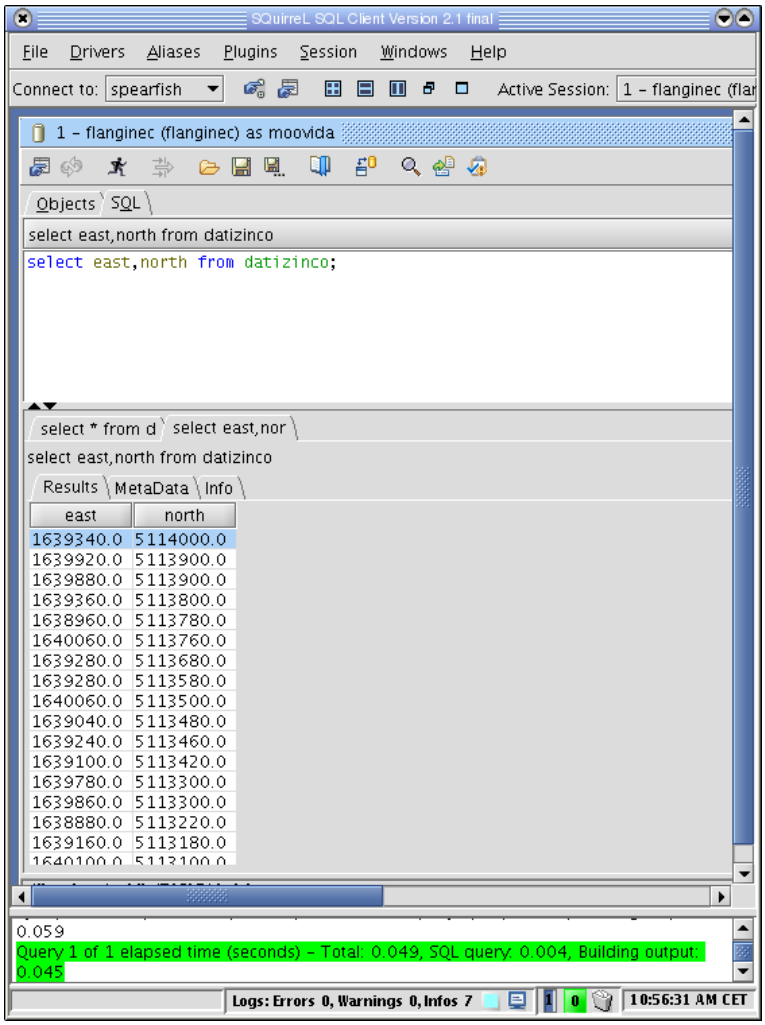

Figure 8.10: Example of a point database query file in SQuirreL.

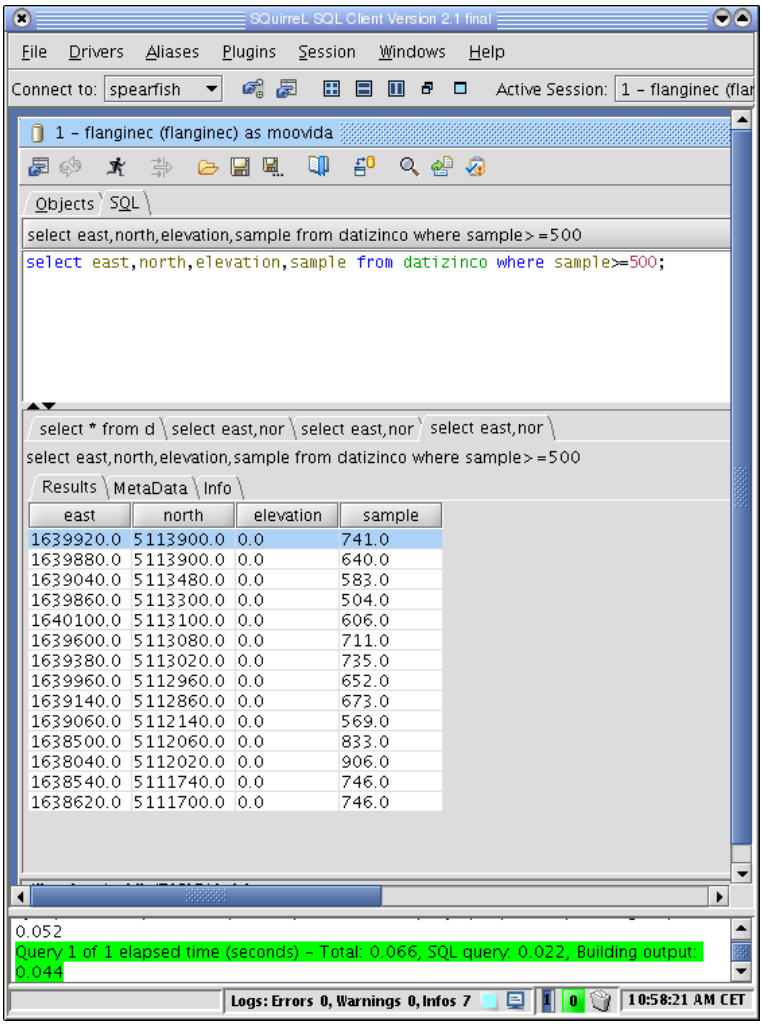
SQuirreL SQL Client Version 2.1 final
File Drivers Aliases Plugins Session Windows Help
Connect to: spearfish
Active Session: 1 - flanginec (flar
1 - flanginec (flanginec) as moovida
Objects SQL
select east,north,elevation,sample from datizinco where sample>=500
select east,north,elevation,sample from datizinco where sample>=500;
select * from d select east,nor select east,nor select east,nor
select east,north,elevation,sample from datizinco where sample>=500
Results MetaData Info
east north elevation sample
1639920.0 5113900.0 0.0 741.0
1639880.0 5113900.0 0.0 640.0
1639040.0 5113480.0 0.0 583.0
1639860.0 5113300.0 0.0 504.0
1640100.0 5113100.0 0.0 606.0
1639600.0 5113080.0 0.0 711.0
1639380.0 5113020.0 0.0 735.0
1639960.0 5112960.0 0.0 652.0
1639140.0 5112860.0 0.0 673.0
1639060.0 5112140.0 0.0 569.0
1638500.0 5112060.0 0.0 833.0
1638040.0 5112020.0 0.0 906.0
1638540.0 5111740.0 0.0 746.0
1638620.0 5111700.0 0.0 746.0
0.052
Query 1 of 1 elapsed time (seconds) - Total: 0.066, SQL query: 0.022, Building output: 0.044
Logs: Errors 0, Warnings 0, Infos 7
10:58:21 AM CET

Figure 8.11: Point database query in SQuirreL.

You can now view the data contained in the database corresponding to the configurations saved in the sql file directly from *JGrass*. To do this you must view the data as points (sites), that is to say click the icon in the map display toolbar. The display wizard integrated into the command provides a first window were you can select the format of the data to be viewed, in this case (Figura 8.12) the data to be viewed is in *sqldatabase* format.

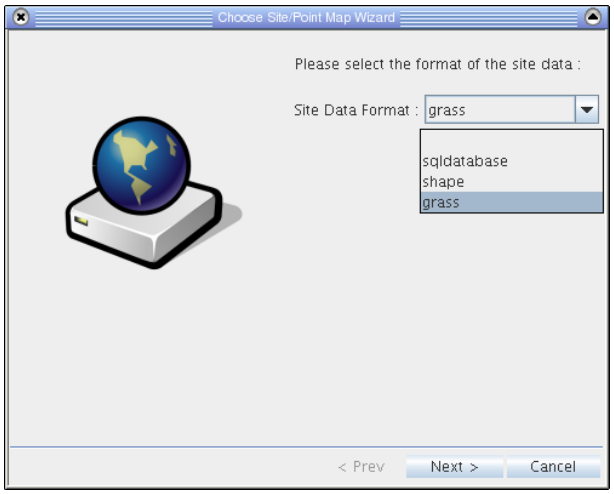

Figure 8.12: The first window of the display wizard to view point data from a PostgreSQL database.

By clicking `Next` you can select the maps to display. Each one of these maps is the result of a specific query to the database, which is saved in the corresponding sql file. This is why in the graphic interface the names of the point maps do not appear but rather the names of the files in the sql directory (figure 8.13).

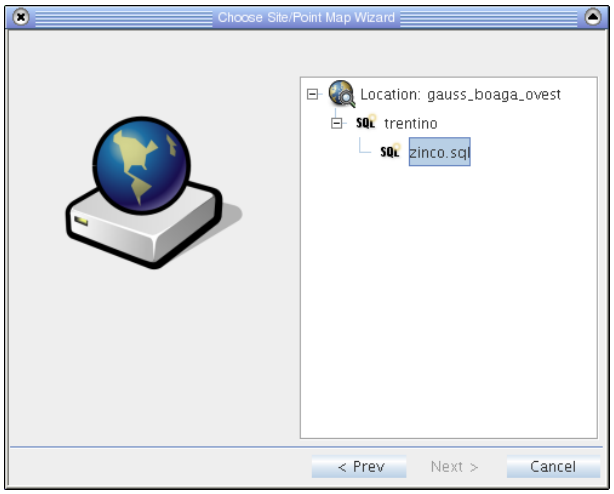

Figure 8.13: Second window of the display wizard to view point data from a PostgreSQL database.

Once you have selected the desired map click `Next` and then `Finish` to display the points. At this point all the processes that can be carried out with *JGrass* sites can also be carried out on the points extracted from a database. Of particular interest are the geostatistical operations which will be presented in following paragraphs.

8.2.2 The Interface for a Spatial Database

A spatial database, as mentioned previously, contains all the information about geometry and about the reference system and cartographic projection. Potentially it can contain all the *features* that a GIS can support: points, lines, polylines, and polygons.

Even here, for a *JGrass* connection to a database of this type, you need to write a connection configuration file. Paradoxically, this file, or rather this type of connection, is much simpler than that used for point data because the spatial database tables have all the information needed for viewing the data. You have only to add a data source called *Postgis*. Postgis refers to visualization of data on a PostgreSQL type database with a Postgis spatial extension.

Before considering the *JGrass* aspect, the data from the Spearfish spatial database is displayed. As can be seen in figure 8.14, in this case, other than the data tables (roads, archsites, bugsites, and streams) there are other tables with spatial information:

- geometry_column: geometric descriptions in binary format of all the geometric shapes contained on the database. All the data in the *the_geom* field of the data table refer to this table.

- spatial_ref_sys: reference system and cartographic projection information

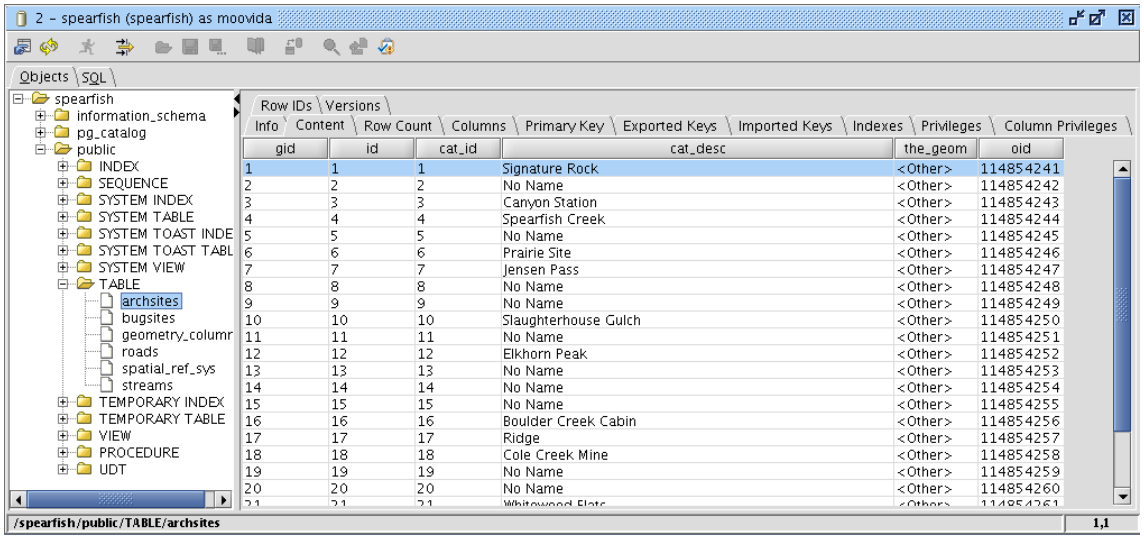

Figure 8.14: Viewing the Spearfish spatial database data in SQuirreL.

In each data table, other than the specific data of the table, there are fields that refer to the geometry and position of each object.

The FRIDA Test Database

An example of connection to a spatial database with different types of objects such as points, lines, and polygons is the free vector dataset called *FRIDA*, it refers to the city of Osnabrueck in Germany.

To view these data you need to create a new workspace that refers to a *location* with the same reference system and cartographic projection as the dataset. The method to create a new workspace is the same as that described in Paragraph 6.1: Creation of a Location. The data is given in the UTM reference system with Potsdam Datum and the Bessel Ellipsoid. Looking through the options of the EPGS file you will find that the reference code is :

31463: `DHDN / 3 - degree Gauss zone 3`

This operation however is not sufficient for defining all the projection parameters. In the following window you will have to insert the region limits and the *towgs84* parameter which are as follows:

- towgs84: 606.0,23.0,413.0
- north: 5801000
- south: 5787000
- east: 3445000
- west: 3427000

To conclude the database connection you will now have to write the configuration file, which has to be saved in the respective directory of the *location*. The file must have an .xml extension and, in particular, it must have the name *postgis.xml*. The file syntax is the same as that used for the point database connection file, but in this case, as shown in figure 8.15, it only has database connection information:

- name of the database
- name of the database on the server
- host address
- username
- password

Once the connection file has been created you can start-up *JGrass* and view the data. In *JGrass*, the FRIDA workspace, created earlier, must be loaded. To view the data on PostgreSQL - Postgis

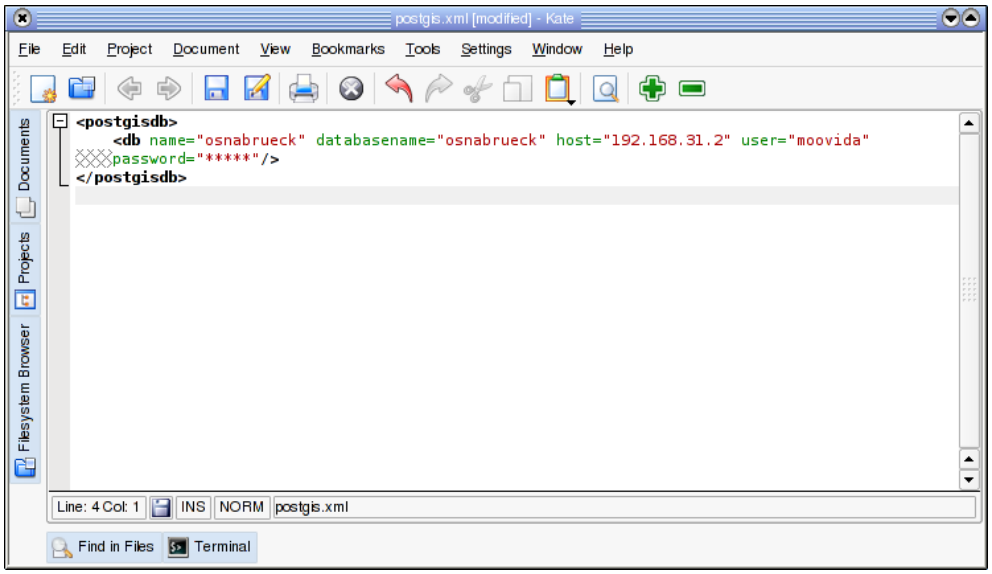

Figure 8.15: Example of the *JGrass* connection file for the Osnabrueck spatial database.

click the icon in the map display toolbar. If the connection configuration file has not been written and saved, an error message will appear (figure 8.16) giving the pathname and the name of the configuration file required to define the database.

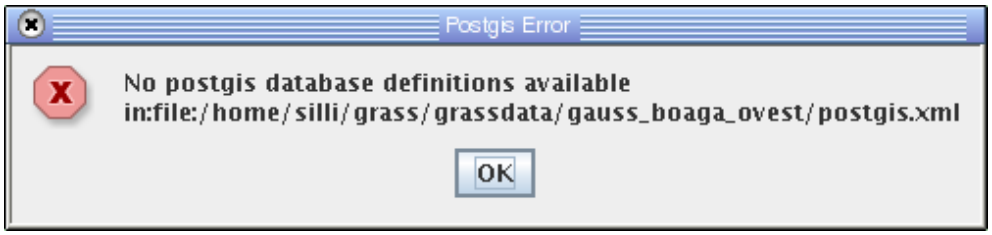

Figure 8.16: Error message for connection to the Postgis spatial database.

If the postgis.xml file has been created correctly and saved in the *JGrass* location directory you can proceed to view the data. The window for viewing Postgis data is very different from the standard *JGrass* window. It allows you to choose the database to connect to and the tables to view (figure 8.17).

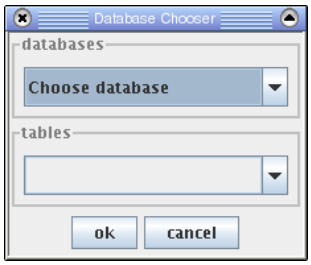

Figure 8.17: First window to view data from a spatial database with Postgis.

By scrolling down the dropdown menus of the two fields, one listing the available databases for the workspace and the other listing the tables therein, you can choose the map to view (figure 8.18).

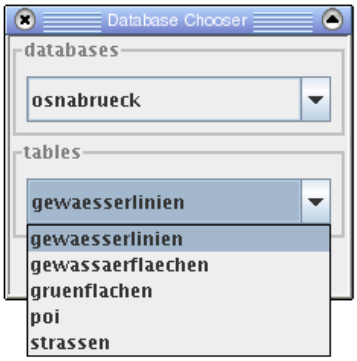

Figure 8.18: Window where to select the map to be viewed from the list of spatial database data.

In this case, you can view polygons, lines, and points, as shown in figure 8.19

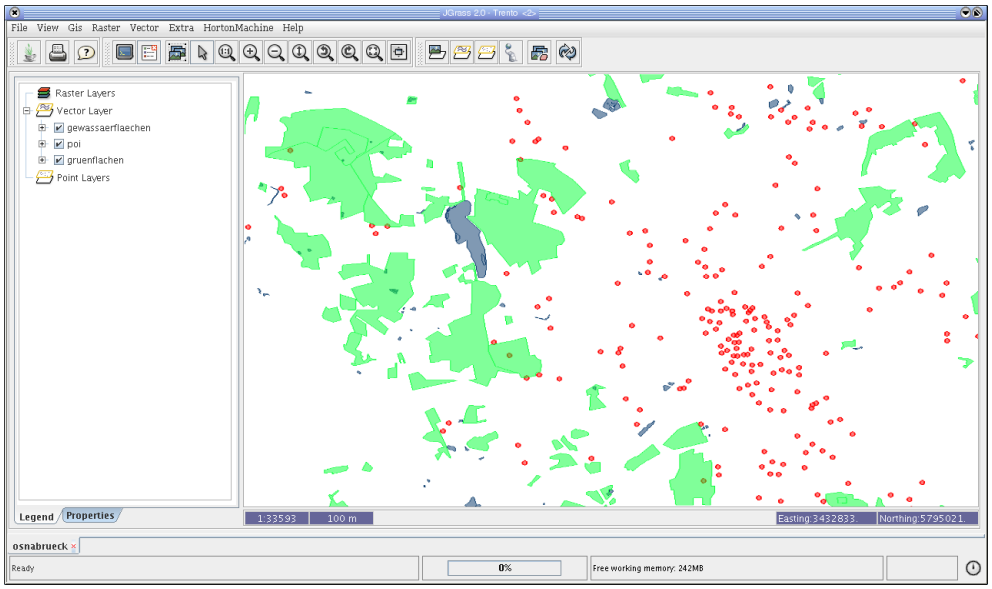

Figure 8.19: Visualization of the data from the FRIDA spatial database.

9 Use of the Console

JGrass, which was originally born as a GRASS graphic interface, proposes to make very advanced commands and algorithms, developed in the scientific field, accessible to the less expert user. On the other hand, *JGrass* is also a development tool for the academic field and, therefore, a research tool. One of the peculiarities of *JGrass* is that of conciliating these two tendencies, furnishing a completely graphic environment and a non-graphic programming environment. The console is the tool that allows you to execute a command or a sequence of commands by inserting all the parameters needed for the operation manually.

The *JGrass* console is not, as of yet, as well structured as the GRASS console but it does provide:

- completion of the command name
- display of the parameters required for an operation
- execution of scripts for the launching of a series of commands

9.1 Execution of a JGrass Command from the Console

To display the console click on the icon, the *JGrass* interface will be modified, displaying the console in the lower part of the screen. The console shown in figure 9.1 is in its standard configuration.

Figure 9.1: The *JGrass* console in its standard configuration.

The console has:

- *prompt*: that is, a symbol that indicates that the console is ready to receive input
- command to enlarge the console
- command to reduce the console
- command to display all the execution information of the commands

When the term `JGrass:` appears in the console it means that it is possible to give a new command, otherwise the console is busy and you will have to wait the end of the execution of the previous command before typing a new one.

If you want to execute a command from the console you need to know, even just indicatively, the name of the command to execute. Even in the *JGrass* console commands can be completed automatically after typing the first few characters of the command name and hitting the *tab* key. For example, if you want to execute the d.rast command you only need to type the the first two characters and then hit *tab* once to view a list of *JGrass* commands that start with the typed characters. Obviously, the more letters you type and the shorter the commands list will be.

To execute a command from the console you need to input all the command parameters (those that are normally input with the graphic interface) preceded by a string that defines it. To know what must be typed at the console in order to execute the command you only need to type the command name followed by the string *–usage*. For example, for the *h.slope* command you should type:

```
JGrass:  h.slope --usage
```

hit enter

```
Usage:  h.slope [--quiet] [--verbose] [--version] [--usage] --pit <pit> --pitmapset
<pitmapset> [--pitformat <pitformat>] --flow <flow> --flowmapset <flowmapset>
[--flowformat <flowformat>] --slope <slope> [--slopemapset <slopemapset>]
[--slopeformat <slopeformat>] [--usegui]
```

The command is one of the geomorphological analysis tools integrated in *JGrass*. It requires as input the depitted digital terrain model (pit), the drainage directions map (flow), as output it gives the slope map (slope). The options in square brackets are not indispensable for the execution of the command and can be omitted, those that are not in square brackets must be specified.

Some options are particular and are present for all the commands:

- *–quiet* allows you to follow the command progress, actually this option is not very significant in *JGrass*
- *–verbose* shows you all the progress of the command and the internal console messages, this option can also be activated with the console information icon
- *–usage* displays the option string shown above
- *–usegui* launches the graphic interface directly from the console, it is the equivalent of clicking directly on the command icon.

Therefore, to execute the h.slope command from the console you have to type:

```
h.slope --pit pit --pitmapset trentino --flow flow --flowmapset trentino --slope
slope --slopemapset trentino
```

and to view the newly created map the command is:

```
d.rast slope
```

9.2 Scripting

The scripting language used by *JGrass* is the Java Beanshell language. BeanShell is a small, free, embeddable, Java source interpreter with object scripting language features, written in Java. BeanShell executes standard Java statements and expressions, in addition to obvious scripting commands and syntax. BeanShell supports scripted objects as simple method closures like those in Perl and JavaScript(tm).

9.3 Execution of GRASS Commands from the Console

The native GRASS commands that have not yet been provided with a graphic interface can be executed directly from the console. The execution mode is the same as that used in GRASS, if you do not know the syntax you only need to type the command and hit enter, the respective page from the GRASS manual will be displayed on the console.

Unfortunately, for native GRASS commands the command completion feature does not work, you must therefore at least know the command name. Typing the command name will either execute it directly or call its help page.

All the GRASS commands that do not require user interaction can be executed in *JGrass*. This means that all the raster and vector commands which require parameter definitions at the command definition stage can be executed. If the command output is a map or a series of maps they are created in the current mapset, if the output is a series of information, these are displayed in the console.

Potentially all the GRASS commands can be executed in *JGrass*, however not all the commands have been tested and classified as interactive and non-interactive yet, and the check-result commands have not yet been tried. The following paragraph gives the complete list of GRASS commands[1] with a general explanation for each command. The testing and checking stage is not

[1] http://www.gdf-hannover.de/media.php?id=7&lg=en

yet complete and so this list, as of yet, exists only in English. These commands are surely all active for users of *JGrass* under Linux e MacOsX, while some could be missing for Windows users (in particular those for the analysis of satellite data, identified with the letter *i.**). To check if the command is available under Windows you only need to search for the executable in the *grasscommands* directory in the *JGrass* installation directory.

9.4 Index of GRASS Commands

The following index is a complete list of all GRASS-commands. An up-to-date list can be found at the official GRASS-GIS website at under the section manuals.

g.* commands:

Command	***Description***
g.access	
g.ask	Prompts the user for the names of GRASS data base files.
g.copy	Copies available data files in the user's current mapset search path and location to the appropriate element directories under the user's current mapset.
g.filename	Prints GRASS data base file names.
g.findfile	Searches for GRASS data base files and sets variables for the shell.
g.gisenv	Outputs the user's current GRASS variable settings.
g.list	Lists available GRASS data base files of the user-specified data type to standard output.
g.manual	display the HTML man pages of GRASS
g.mapset	Change current mapset
g.mapsets	Modifies the user's current mapset search path, affecting the user's access to data existing under the other GRASS mapsets in the current location.
g.mlist	Apply regular expressions and wildcards to g.list
g.mremove	Apply regular expressions and wildcards to g.remove
g.parser	***g.parser***
g.proj	Prints and manipulates GRASS projection information files.
g.region	Program to manage the boundary definitions for the geographic region.
g.remove	Removes data base element files from the user's current mapset.
g.rename	To rename data base element files in the user's current mapset.
g.setproj	***g.setproj***
g.tempfile	Creates a temporary file and prints the file name.
g.version	Displays version and copyright information.

i.* commands:

Command	*Description*
i.cca	Canonical components analysis (cca) program for image processing.
i.class	***i.class***
i.cluster	An imagery function that generates spectral signatures for land cover types in an image using a clustering algorithm. The resulting signature file is used as input for i.maxlik, to generate an unsupervised image classification.

Prosecution on next side

Prosecution

Command	*Description*
i.fft	Fast Fourier Transform (FFT) for image processing.
i.fusion.brovey	Brovey transform to merge multispectral and high-res panchromatic channels
i.gensig	Generates statistics for i.maxlik from raster map layer.
i.gensigset	Generate statistics for i.smap from raster map layer.
i.group	Creates and edits groups and subgroups of imagery files.
i.his.rgb	Hue-intensity-saturation (his) to red-green-blue (rgb) raster map color transformation function.
i.ifft	Inverse Fast Fourier Transform (ifft) for image processing.
i.image.mosaic	Mosaics up to 4 images and extends colormap; creates map *.mosaic
i.maxlik	An imagery function that classifies the cell spectral reflectances in imagery data based on the spectral signature information generated by either i.cluster, i.class, or i.gensig.
i.oif	Calculates Optimum-Index-Factor table for LANDSAT TM bands 1-5, & 7
i.ortho.photo	***i.ortho.photo***
i.pca	Principal components analysis (pca) program for image processing.
i.points	
i.rectify	Rectifies an image by computing a coordinate transformation for each pixel in the image based on the control points
i.rgb.his	Red-green-blue (rgb) to hue-intensity-saturation (his) raster map color transformation function.
i.smap	Performs contextual image classification using sequential maximum a posteriori (SMAP) estimation.
i.spectral	displays spectral response at user specified locations in images
i.target	Targets an imagery group to a GRASS location and mapset.
i.tasscap	Tasseled Cap (Kauth Thomas) transformation for LANDSAT-TM data
i.vpoints	***i.vpoints***
i.zc	Zero-crossing “edge detection” raster function for image processing.

p.* commands:

Command	*Description*
p.out.vrml	module to output GRASS data in the format of Virtual Reality Modeling Language (VRML)

pg.* commands:

Command	*Description*
pg.postgisdb	***pg.postgisdb***

photo.* commands:

Command	*Description*
photo.2image	***photo.2image***
photo.2target	***photo.2target***
photo.camera	
photo.init	***photo.init***
photo.rectify	***photo.rectify***

ps.* commands:

Command	*Description*
ps.map	Hardcopy PostScript map output utility.

r.* commands:

Command	*Description*
r.average	Finds the average of values in a cover map within areas assigned the same category value in a user-specified base map.
r.basins.fill	Generates a raster map layer showing watershed subbasins.
r.bilinear	Bilinear interpolation utility for raster map layers.
r.blend	Blends color components of 2 raster maps by a given % first map'
r.buffer	Creates a raster map layer showing buffer zones surrounding cells that contain non-NULL category values.
r.cats	Prints category values and labels associated with user-specified raster map layers.
r.circle	Creates a raster map containing concentric rings around a given point.
r.clump	Recategorizes data in a raster map layer by grouping cells that form physically discrete areas into unique categories.
r.coin	Tabulates the mutual occurrence (coincidence) of categories for two raster map layers.
r.colors	Creates/Modifies the color table associated with a raster map layer.
r.composite	Combines red, green and blue map layers into a single composite map layer.
r.compress	Compresses and decompresses raster files.
r.contour	Produces a GRASS binary vector map of specified contours from GRASS raster map layer.
r.cost	Outputs a raster map layer showing the cumulative cost of moving between different geographic locations on an input raster map layer whose cell category values represent cost.

Prosecution on next side

Prosecution

Command	***Description***
r.covar	Outputs a covariance/correlation matrix for user-specified raster map layer(s).
r.cross	Creates a cross product of the category values from multiple raster map layers.
r.describe	Prints terse list of category values found in a raster map layer.
r.digit	***r.digit***
r.distance	Locates the closest points between objects in two raster maps.
r.drain	Traces a flow through an elevation model on a raster map layer.
r.fill.dir	Filters and generates a depressionless elevation map and a flow direction map from a given elevation layer
r.fillnulls	Fills no-data areas in raster maps using v.surf.rst splines interpolation
r.flow	Construction of slope curves (flowlines), flowpath lengths, and flowline densities (upslope areas) from a raster digital elevation model(DEM).
r.grow	Generates a raster map layer with contiguous areas grown by one cell.
r.his	Generates red, green and blue raster map layers combining hue, intensity, and saturation (his) values from user-specified input raster map layers.
r.in.arc	Convert an ESRI ARC/INFO ascii raster file (GRID) into a (binary) raster map layer.
r.in.ascii	Convert an ASCII raster text file into a (binary) raster map layer.
r.in.bin	Import a binary raster file into a GRASS raster map layer.
r.in.gdal	Import GDAL supported raster file into a binary raster map layer.
r.in.gridatb	Imports GRIDATB.FOR map file (TOPMODEL) into GRASS raster map
r.in.mat	Import a binary MAT-File(v4) to a GRASS raster.
r.in.poly	Create raster maps from ascii polygon/line data files in the current directory.
r.in.srtm	Import SRTM90 HGT files into GRASS
r.info	Outputs basic information about a user-specified raster map layer.
r.kappa	Calculate error matrix and kappa parameter for accuracy assessment of classification result.
r.le.patch	
r.le.pixel	
r.le.setup	***r.le.setup***
r.le.trace	
r.los	Line-of-sight raster analysis program.
r.mapcalc	***r.mapcalc***
r.mapcalculator	r.mapcalculator - Calculates new raster map from r.mapcalc expression
r.median	Finds the median of values in a cover map within areas assigned the same category value in a user-specified base map.
r.mfilter	Raster file matrix filter.

Prosecution on next side

Prosecution

Command	***Description***
r.mode	Finds the mode of values in a cover map within areas assigned the same category value in a user-specified base map.
r.neighbors	Makes each cell category value a function of the category values assigned to the cells around it, and stores new cell values in an output raster map layer.
r.null	The function of r.null is to explicitly create the NULL-value bitmap file.
r.out.arc	Converts a raster map layer into an ESRI ARCGRID file.
r.out.ascii	Converts a raster map layer into an ASCII text file.
r.out.bin	Exports a GRASS raster to a binary array.
r.out.gdal	Exports GRASS raster data into various formats (requires GDAL)
r.out.gridatb	Exports GRASS raster map to GRIDATB.FOR map file (TOPMODEL)
r.out.mat	Exports a GRASS raster to a binary MAT-File.
r.out.mpeg	Raster File Series to MPEG Conversion Program.
r.out.png	Export GRASS raster as non-georeferenced PNG image format.
r.out.pov	Converts a raster map layer into a height-field file for POVRAY.
r.out.ppm	Converts a GRASS raster file to a PPM image file at the pixel resolution of the CURRENTLY DEFINED REGION.
r.out.ppm3	Converts 3 GRASS raster layers (R,G,B) to a PPM image file at the pixel resolution of the CURRENTLY DEFINED REGION.
r.out.tiff	Exports a GRASS raster file to a 8/24bit TIFF image file at the pixel resolution of the currently defined region.
r.param.scale	Extracts terrain parameters from a DEM. Uses a multi-scalar approach by taking fitting quadratic parameters to any size window (via least squares)
r.patch	Creates a composite raster map layer by using known category values from one (or more) map layer(s) to fill in areas of "no data" in another map layer.
r.plane	Creates raster plane map given dip (inclination), aspect (azimuth), and one point
r.profile	Outputs the raster map layer values lying on user-defined line(s).
r.proj	Re-project a raster map from one location to the current location.
r.quant	This routine produces the quantization file for a floating-point map.
r.random.cells	Generates random cell values with spatial dependence.
r.random	Creates a raster map layer and vector point map containing randomly located sites.
r.random.surface	Generates random surface(s) with spatial dependence.
r.reclass.area	Reclasses a raster map greater or less than user specified area size (in hectares)
r.reclass	Creates a new map layer whose category values are based upon the user's reclassification of categories in an existing raster map layer.
r.recode	Recode raster maps.

Prosecution on next side

Prosecution

Command	*Description*
r.region	Sets the boundary definitions for a raster map.
r.regression.line	Calculates linear regression from two raster maps: y = a + b*x
r.report	Reports statistics for raster map layers.
r.resamp.rst	Reinterpolates and computes topographic analysis from input raster file to a new raster file (possibly with different resolution) using regularized spline with tension and smoothing.
r.resample	GRASS raster map layer data resampling capability.
r.rescale.eq	Rescales histogram equalized the range of categoryvalues in a raster map layer.
r.rescale	Rescales the range of category values in a raster map layer.
r.ros	Generates three, or four raster map layers showing 1) the base (perpendicular) rate of spread (ROS), 2) the maximum (forward) ROS, 3) the direction of the maximum ROS, and optionally 4) the maximum potential spotting distance.
r.series	Makes each output cell value a function of the values assigned to the corresponding cells in the input raster map layers.
r.shaded.relief	Creates shaded relief map from an elevation map (DEM).
r.slope.aspect	Generates raster map layers of slope, aspect, curvatures and partial derivatives from a raster map layer of true elevation values. Aspect is calculated counterclockwise from east.
r.spread	Simulates elliptically anisotropic spread on a graphics window and generates a raster map of the cumulative time of spread, given raster maps containing the rates of spread (ROS), the ROS directions and the spread origins. It optionally produces raster maps to contain backlink UTM coordinates for tracing spread paths.
r.spreadpath	Recursively traces the least cost path backwards to cells from which the cumulative cost was determined.
r.statistics	Category or object oriented statistics.
r.stats	Generates area statistics for raster map layers.
r.sum	Sums up the raster cell values.
r.sun	Computes direct (beam), diffuse and reflected solar irradiation raster maps for given day, latitude, surface and atmospheric conditions. Solar parameters (e.g. sunrise, sunset times, declination, extraterrestrial irradiance, daylight length) are saved in a local text file. Alternatively, a local time can be specified to compute solar incidence angle and/or irradiance raster maps. The shadowing effect of the topography is optionally incorporated.
r.sunmask	Calculates cast shadow areas from sun position and DEM. Either A: exact sun position is specified, or B: date/time to calculate the sun position by r.sunmask itself.
r.surf.area	Surface area estimation for rasters.
r.surf.contour	Surface generation program from rasterized contours.

Prosecution on next side

Prosecution

Command	***Description***
r.surf.fractal	GRASS module to create a fractal surface of a given fractal dimension. Uses spectral synthesis method. Can create intermediate layers showing the build up of different spectral coefficients (see Saupe, pp.106-107 for an example of this). Use this module to generate naturally looking synthetical elevation models (DEM).
r.surf.gauss	GRASS module to produce a raster map layer of gaussian deviates whose mean and standard deviation can be expressed by the user. It uses a gaussian random number generator.
r.surf.idw	Surface interpolation utility for raster map layers.
r.surf.random	Produces a raster map layer of uniform random deviates whose range can be expressed by the user.
r.terraflow	Flow computation for massive grids (Float version).
r.texture	Generate images with textural features from a raster map
r.thin	Thins non-zero cells that denote linear features in a raster map layer.
r.timestamp	Print/add/remove a timestamp for a raster map.
r.to.vect	Converts a raster map into a vector map layer.
r.topidx	Creates topographic index, ln(a/tan(beta)), map from elevation map.
r.topmodel	Simulates TOPMODEL which is physically based hydrologic model.
r.transect	Outputs raster map layer values lying along user defined transect line(s).
r.univar	Calculates univariate statistics from the non-null cells of a raster map.
r.univar.sh	calculates univariate statistics from a GRASS raster map
r.water.outlet	Watershed basin creation program.
r.watershed	Watershed basin analysis program.
r.what	Queries raster map layers on their category values and category labels.

r3.* commands:

Command	*Description*
r3.in.ascii	Convert a 3D ASCII raster text file into a (binary) 3D raster map layer
r3.in.v5d	import of 3-dimensional Vis5D files (i.e. the v5d file with 1 variable and 1 time step)
r3.info	Outputs basic information about a user-specified 3D raster map layer.
r3.mapcalc	***r3.mapcalc***
r3.mask	Establishes or removes the current working 3D raster mask.
r3.mkdspf	
r3.null	Explicitly create the 3D NULL-value bitmap file.
r3.out.ascii	Converts a 3D raster map layer into an ASCII text file
r3.out.v5d	Export of GRASS 3D raster file to 3-dimensional Vis5D file.
r3.timestamp	print/add/remove a timestamp for a 3D raster map

v.* commands:

Command	*Description*
v.buffer	Create a buffer around features of given type (areas must contain centroid).
v.build.all	***v.build.all***
v.build	Creates topology for GRASS vector data.
v.build.polylines	Build polylines from lines.
v.category	Attach, delete or report vector categories to map geometry.
v.clean	Toolset to clean vector topology.
v.convert.all	
v.convert	Imports older versions of GRASS vectors.
v.db.connect	prints/sets DB connection for a vector map
v.db.select	Print vector attributes
v.delaunay	Create a Delaunay triangulation from an input vector of points or centroids.
v.digit	
v.distance	Find the nearest element in vector 'to' for elements in vector 'from'. Various information about this relation may be uploaded to the attribute table of input vector 'from' or printed to stdout.
v.external	Create a new vector as a read-only link to OGR layer. Available drivers:
v.extract	Selects vector objects from an existing vector map and creates a new map containing only the selected objects. If 'list', 'file' and 'where' options are not specified, all features of given type and layer are extracted, categories are not changed in that case.
v.hull	Uses a GRASS vector points map to produce a convex hull vector map
v.in.ascii	Convert GRASS ascii file or points file to binary vector.

Prosecution on next side

Prosecution

Command	***Description***
v.in.db	Create new vector (points) from database table containing coordinates.
v.in.e00	Import of E00 file into a vector map.
v.in.garmin	Upload Waypoints, Routes, and Tracks from a Garmin GPS receiver into a vector map.
v.in.ogr	Convert OGR vectors to GRASS.
v.in.region	Create a new vector from current region.
v.in.sites.all	
v.in.sites	Converts a GRASS site_lists file into a vector file.
v.info	Outputs basic information about a user-specified vector map layer.
v.kcv	Randomly partition points into test/train sets.
v.kernel	Generates a raster density map from vector points data using a moving 2D isotropic Gaussian kernel or optionally generates a vector density map on vector network with a 1D kernel
v.label	Create paint labels for GRASS vector file and attached attributes.
v.mkgrid	Creates a (binary) GRASS vector map of a user-defined grid.
v.neighbors	Makes each cell value a function of the attribute values assigned to the vector points or centroids around it, and stores new cell values in an output raster map layer.
v.net.alloc	Allocate subnets for nearest centres (direction from centre). Centre node must be opened (costs >= 0). Costs of centre node are used in calculation
v.net	Network maintenance.
v.net.iso	Split net to bands between cost isolines (direction from centre). Centre node must be opened (costs >= 0). Costs of centre node are used in calculation
v.net.path	Find shortest path on vector network. Reads start/end pointsfrom standard input in 2 possible formats:
v.net.salesman	Create a cycle connecting given nodes (Traveling salesman problem). Note that TSP is NP-hard, heuristic algorithm is used by this module and created cycle may be sub optimal.
v.net.steiner	Create Steiner tree for the network and given terminals. Note that 'Minimum Steiner Tree' problem is NP-hard and heuristic algorithm is used in this module so the result may be sub optimal.
v.normal	tests for normality for points.
v.out.ascii	Convert a GRASS binary vector map to a GRASS ASCII vector map
v.out.dxf	Exports GRASS vector files to DXF file format.
v.out.ogr	Convert to OGR format.
v.out.pov	Convert to POV-Ray format, GRASS x,y,z -> POV-Ray x,z,y
v.overlay	Overlay 2 vector maps.
v.patch	Creates a new binary vector map layer by combining other binary vector map layers.

Prosecution on next side

Prosecution

Command	***Description***
v.perturb	Random location perturbations of GRASS sites.
v.proj	Allows projection conversion of vector files.
v.qcount	indices for quadrat counts of sites lists
v.random	Randomly generate a GRASS vector points map.
v.reclass	Changes vector category values for an existing vector map according to results of SQL queries or a value in attribute table column.
v.sample	Sample a raster file at site locations.
v.segment	Create points/segments from input lines, and positions read from stdin in format:
v.select	Select features from ainput by features from binput.
v.surf.idw	Surface interpolation from sites data by Inverse Distance Squared Weighting.
v.surf.rst	Interpolation and topographic analysis from given point or contour data in vector format to GRASS floating point raster format using regularized spline with tension.
v.to.db	Load values from vector to database. In uploaded/printed category values '-1' is used for 'no category' and 'null'/'-' if category cannot be found or multiple categories were found.
v.to.points	Create points along input lines.
v.to.rast	Converts a binary GRASS vector map layer into a GRASS raster map layer.
v.transform	Transforms an vector map layer from one coordinate system into another coordinate system.
v.type	Change the type of geometry elements.
v.univar	Calculates univariate statistics for attribute. Variance and standard deviation is calculated only for points.
v.vol.rst	Interpolates point data to a G3D grid volume using regularized spline with tension (RST) algorithm
v.voronoi	Create a Delaunay triangulation from an input vector of points or centroids.
v.what.rast	Upload raster values at positions of vector points to the table.

10 Statistical Processing

Some statistical processing commands have been implemented in *JGrass* through its connection to the Open Source statistical calculation software *R*. These commands are found in:

- r.in.lidar: referring to the importation of point triplets and the interpolation of the data in the z column for the creation of rasters.
- g.statistics: some commands for the statistical processing of point maps

10.1 Statistical Processing of Point Maps

The *g.statistics* command allows you to view statistical reports of the significant data of a data set such as the mean, the median, the variance, and the maximum and minimum values, and to calculate and graph the probability density functions and the cumulative probability functions. The most important algorithm of this set of commands is surely, however, the command that allows you to carry out statistical interpolation by means of the *kriging* geostatistical interpolator. The graphic interface of this command is shown in figure 10.1.

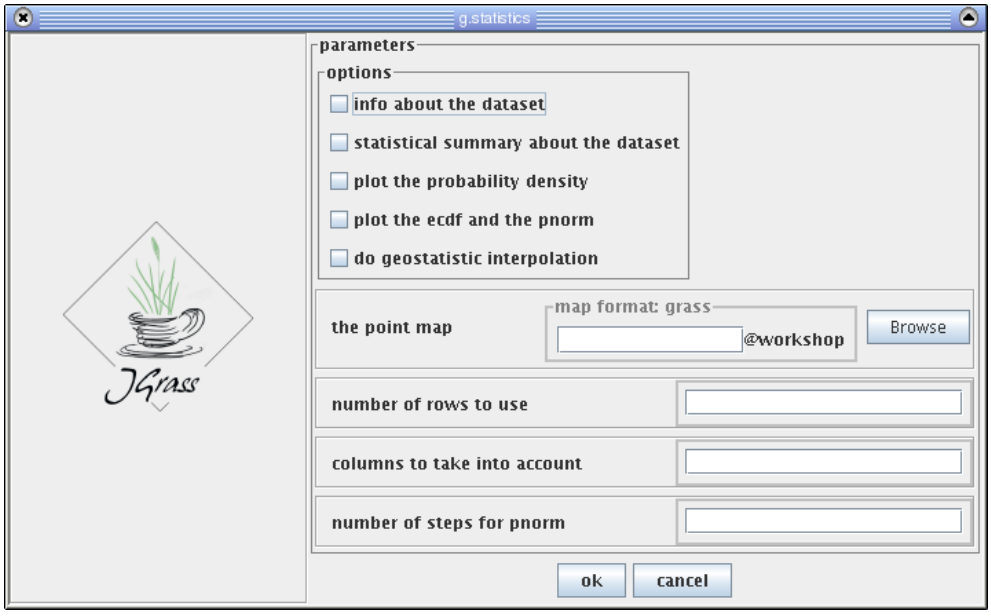

Figure 10.1: The graphic interface of the *g.statistics* command.

The parameters that can be set are:

1. parameters to be calculated (in the upper part of the widow)

2. point map to be processed
3. number of rows to view in the graphic report
4. the column to take into account in statistical calculations in the case that various attributes have been specified
5. number of intervals in which to divide the complete range of data for the display of the probability density function

As has already been mentioned, the command only works with point maps that have been imported to *JGrass* as *sites*. To do this you only need to write a file in the format described below and save it to the *site_lists* directory of the working mapset. The file should not have an extension and it must be written with a header that contains the name of the file and a description of the what the file contains, and an entire section dedicated to the point data: x - y coordinates and the various attributes. The column separator is the — character, as shown below:

```
name|sites_flanginec
desc|Cd at ground
1639705|5114025 |#8.2 @station1
1639355|5114005 |#2.4 @station2
1639485|5113725 |#3.0 @station3
1638745|5113585 |#1.6 @station4
1639105|5113505 |#4.2 @station5
```

To view a point map you only need to choose the *Display a site map* option from the *View* menu or click on the icon. The window shown in figure 10.2 will appear, where it is possible to select the point map format to be displayed. As a general rule select the GRASS format insomuch that the other formats are in substitution phase with the relevant commands. Click *Next* to continue.

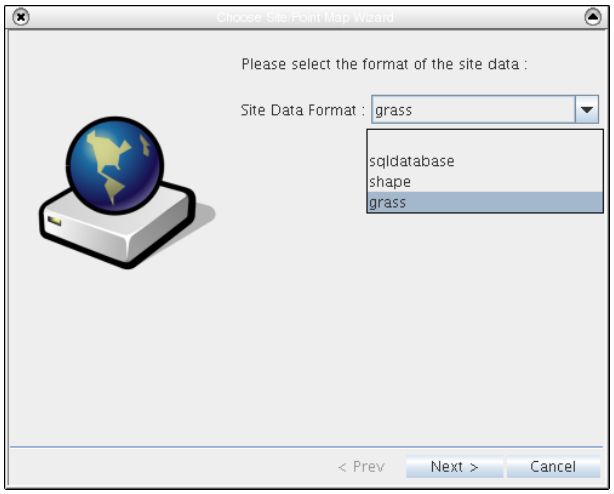

Figure 10.2: First window of the site/point map wizard in *JGrass*.

The second window (figure 10.3) allows you to select the mapset where the data to be displayed is found, in the example the *trentino* mapset has been selected, it is the current mapset where the file with point data has been saved. Click *Next* to continue.

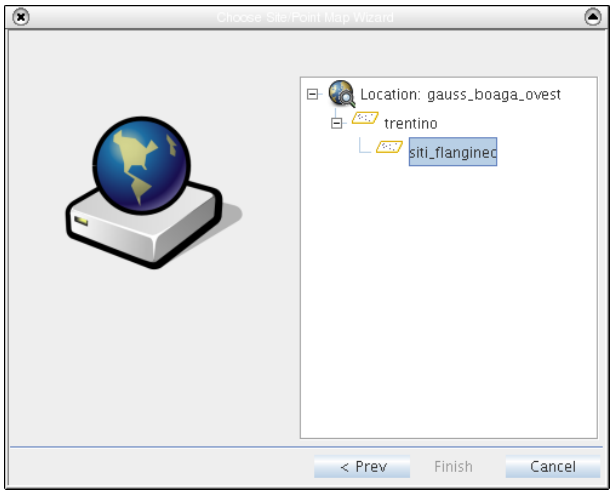

Figure 10.3: Second window of the site/point map wizard in *JGrass*.

The next window allows you select the point display options, the character used, the colour, and the dimensions of the point. Select the desired configurations and click *Finish*.

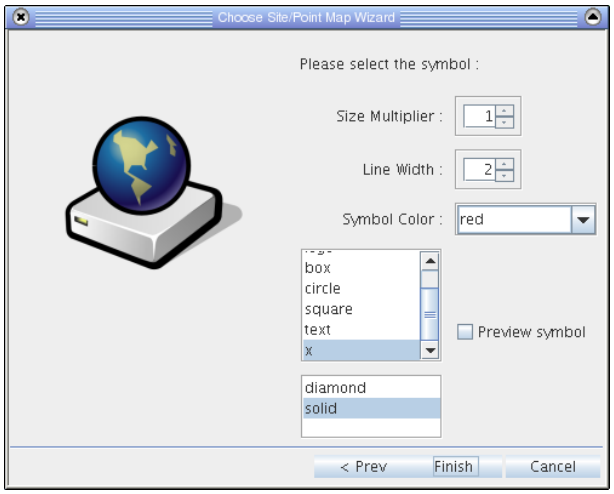

Figure 10.4: Third window of the site/point map wizard in *JGrass*.

The resulting point map, superimposed on the DEM is shown in figure 10.5, the display colour of the points was changed by right-clicking on the map name in the *legend* window and selecting *proprieties* from the popup menu.

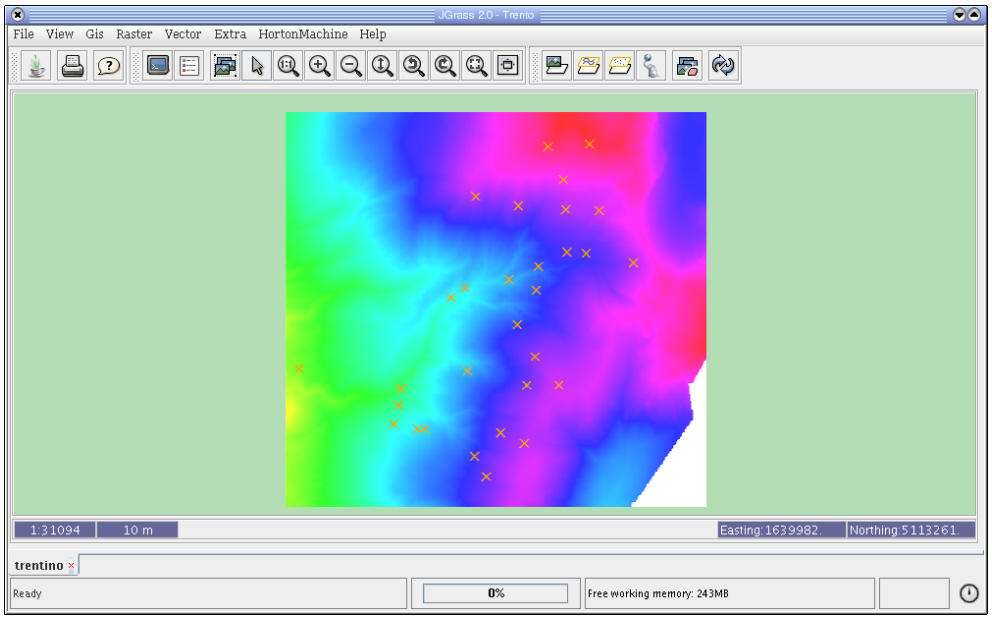

Figure 10.5: Visualization of a point map in GRASS format in *JGrass*.

Once you have the point map you can carry out the *g.statistics* command with its various options.

10.2 Statistical Report of the Data Set

You can view a statistical report, as well as a selection of data from the file (specifying the number of rows to be displayed in the graphic interface), by checking the first two options of the command interface (figure 10.6).

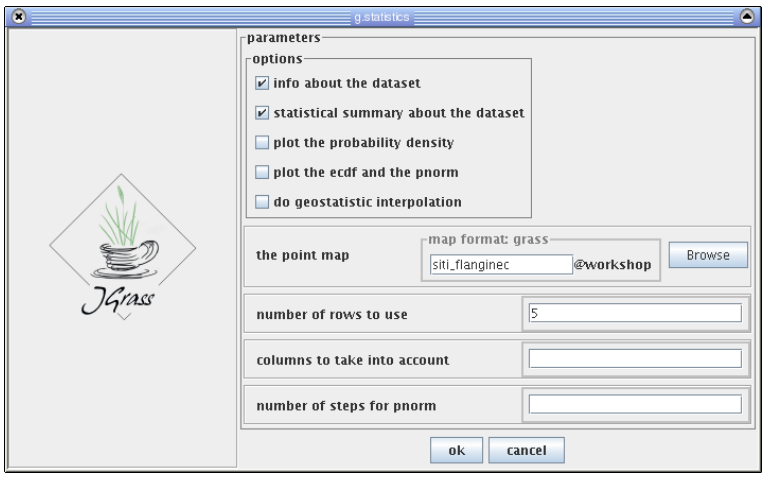

Figure 10.6: Graphic interface of the *JGrass g.statistics* command requiring a statistical report.

The so called statistical report gives the more important data of the data set contained in the map (figure 10.7):

- number of data
- minimum value
- 25^th percentile
- median
- mean
- 75^th percentile
- maximum value
- variance

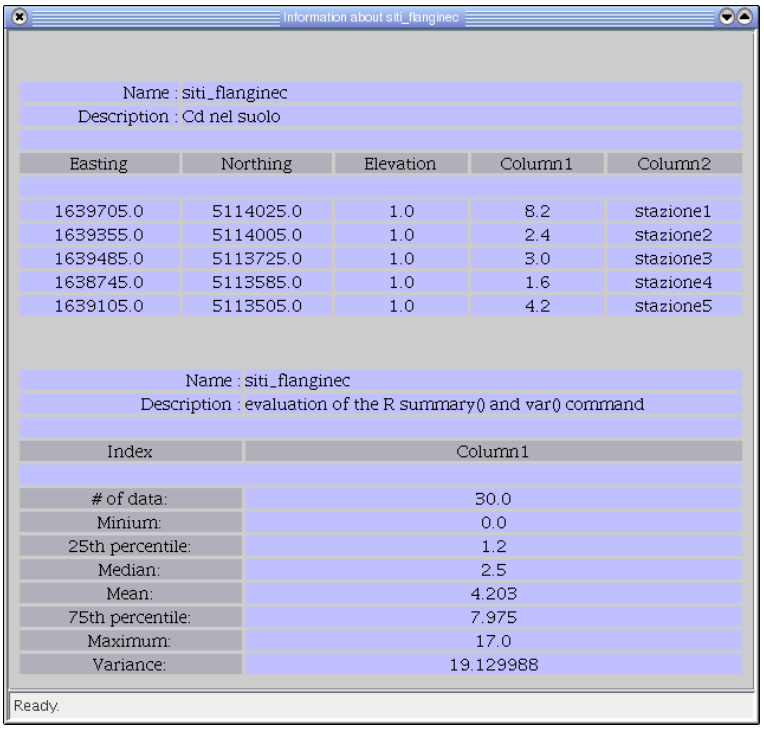

Figure 10.7: The html report of the *JGrass g.statistics* command for the display of the statistical report.

10.3 The Probability Density Function

The probability density function of the data in the map is, by definition, a function that expresses the probability that the value assumed by the casual variable `Z` falls within the interval defined by `z` e `(z+dz)` through the relation:

$$P(z \min Z \min z + dz) = f(z) \cdots dz$$

To plot the function you must check the third option of those given (see figure 10.8).

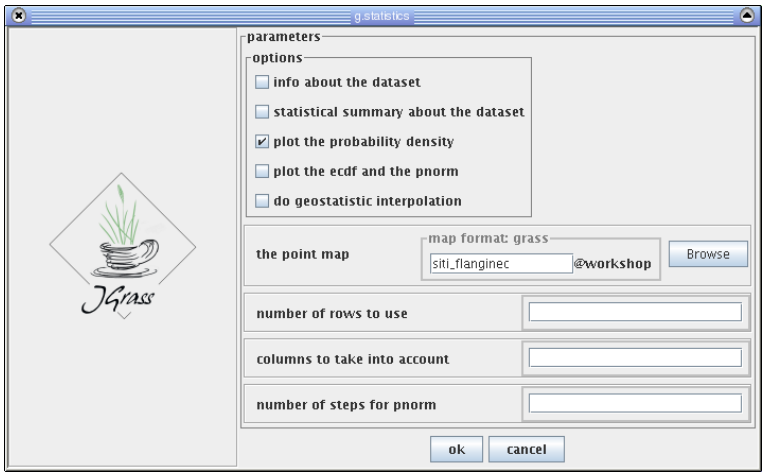

Figure 10.8: Graphic interface of the *JGrass g.statistics* command requiring the probability density function to be plotted.

The result is shown in figure 10.9.

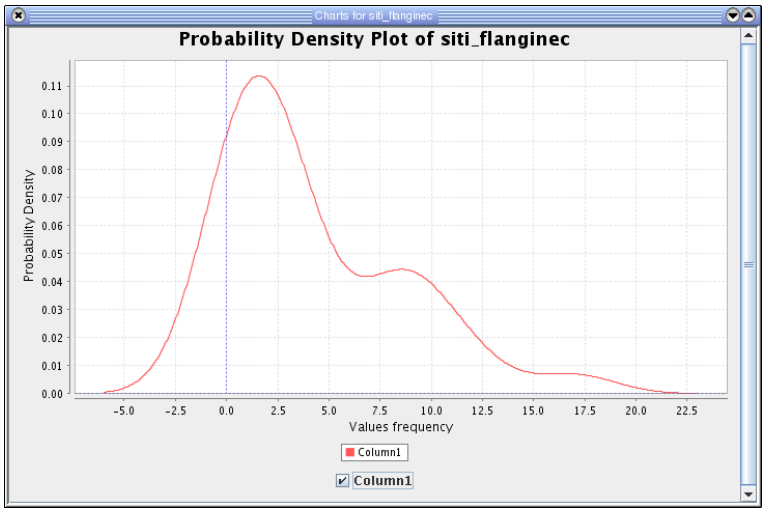

Figure 10.9: Result of the *JGrass g.statistics* command for the probability density function.

10.4 The Cumulative Distribution Function

The algorithm that calculates the cumulative probability function and its interpolation with the standard normal model (Gaussian model) has also been implemented in *JGrass*. By definition the cumulative probability function is the function that expresses the portion of data that does not exceed an assigned threshold value z.

$$F(z) = P(Z \leq z)$$

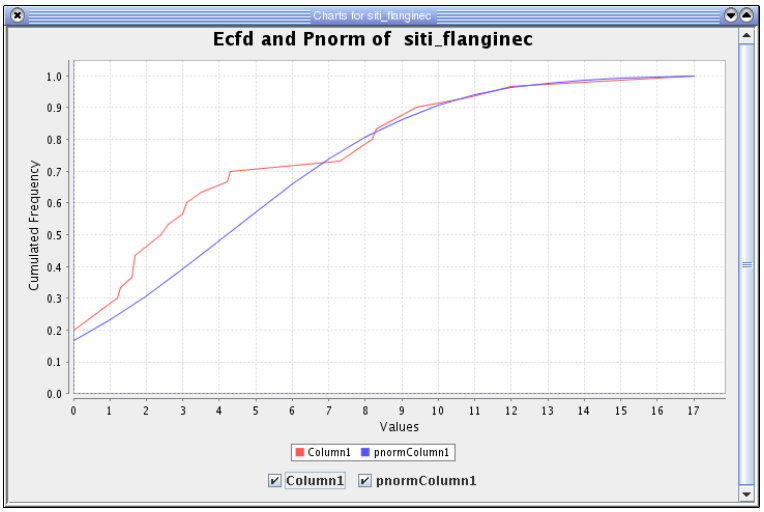

Figure 10.10: Result of the *JGrass g.statistics* command for the cumulative probability function.

10.5 Geostatistical Interpolation with the Kriging Interpolator

In *JGrass*, the kriging point data interpolation algorithm has been implemented by means of the connection with *R*. Kriging is a geostatistical interpolator, it is a least-squares linear regression algorithm which, on the basis of known values, calculates the values of the variable in the other points of a regular grid.

The spatial correlation of the data is expressed on the basis of available measures and specific models which estimate the value of a continuous variable in non-measurement points. This estimate is obtained by means of interpolation of the available data.

To have an idea of the correlation between available measures, or rather of the spatial variability of the measure, the concept of the semivariogram is used. The semivariogram usually presents a tendency to grow with the distance, reflecting a decrease in correlation with distance between pairs of points. It represents the mean difference of the value of two points on interpolating surface at a distance *h*. There are many types of semivariogram that can be adapted to the real one, the main ones are listed in figure 10.11.

If you use the *g.statistics* command to interpolate a set of point data, the first thing to do is create the semivariogram of the real data and find the model that best adapts to the specific case. Once you have defined the semivariogram model to use you can proceed with the data interpolation. Therefore, check the *do geostatistic interpolation* option and select the point map to interpolate, then click *ok*. The window shown in figure 10.12 will appear, the number of intervals and the maximum correlation distance in metres are required. Click *ok* to continue with the calculations.

Model	Expression	Usage
Nugget	$\gamma(h)=\begin{cases} h=0 \rightarrow 0 \\ h\neq 0 \rightarrow 1 \end{cases}$	The phenomena are spatially uncorrelated
Spheric	$\gamma(h)=\begin{cases} h\leq a \rightarrow 1.5\frac{h}{a}-0.5\left(\frac{h}{a}\right)^3 \\ h>a \rightarrow 1 \end{cases}$	Stationary periodic processes with a period "a".
Gaussian	$\gamma(h)=1-\exp\left(\frac{-3h}{a}\right)$	Estimating this range, all the scales present in the phenomena are revealed
Exponential	$\gamma(h)=1-\exp\left(\frac{-3h^2}{a^2}\right)$	
Power Law	$\gamma(h)=h^{\omega};0<\omega<2$	Non stationary phenomena (mean and variance are not spatially constant)

Figure 10.11: Table of the formulae and typology of use of the main types of semivariogram usually used in geostatistics.

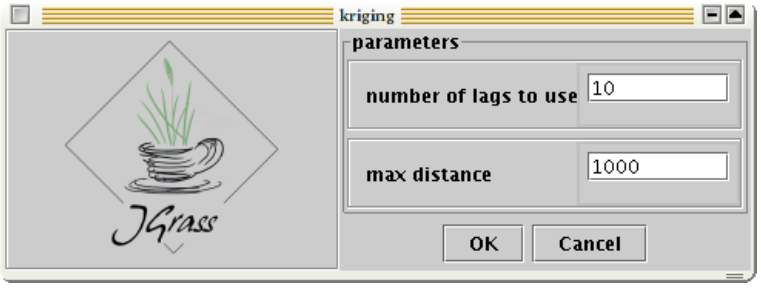

Figure 10.12: Second window of the procedure for the geostatistic interpolation of point data.

At the end of the process a window will inform you of the number of bins that were actually created and the greatest distance found between points. It is very important that all the intervals into which it was decided to divide the data set (figure 10.12) be actually created, otherwise there will be an error during the procedure and the *R* connection will crash.

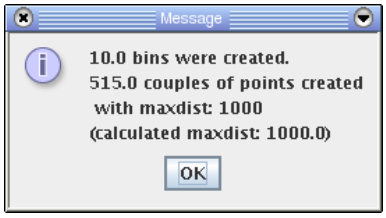

Figure 10.13: Message informing of the subdivision of the data set into intervals for the geostatistic interpolation.

To continue you only need to confirm this data subdivision and the graph shown in figure 10.14 will be displayed, that is the semivariogram calculated from real data.

To continue the procedure just close the graph window, a dialog box will appear requiring you to select the model and parameters in order to best represent the calculated semivariogram (figure

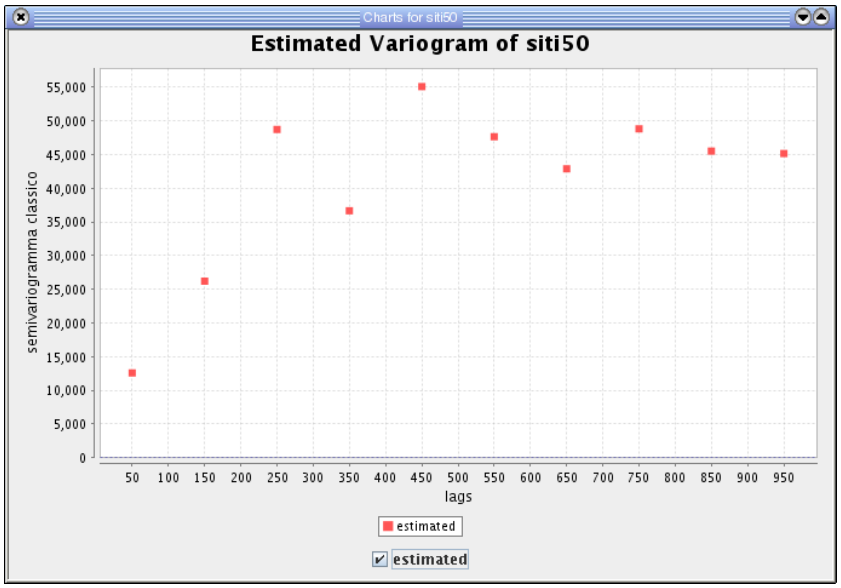

Figure 10.14: Variogram of the data contained in the point map.

10.15). In this way a mathematical formula is assigned to the semivariogram which will be used in the actual interpolation.

Figure 10.15: Dialog box for the selection of the semivariogram model that best describe the real semivariogram and for the definition of the Gaussian model parameters.

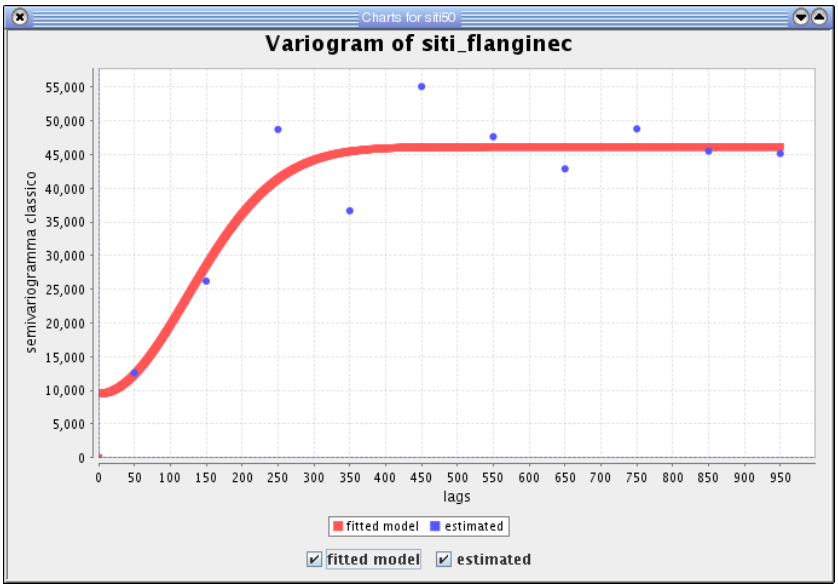

Figure 10.16: Graph with the superimposition of the real semivariogram and the one obtained with the Gaussian model using the parameters defined in the interface of figure 10.15.

To continue just close the graph window, a message will appear asking you if you wish to continue with the interpolation procedure or if you wish to redefine the model parameters in order to better represent the real data.

Figure 10.17: Message window asking to continue the interpolation procedure and create an interpolated map from the point map.

The same procedure can be repeated with the exponential model, inputting different parameter values on the basis of the diagram of figure 10.18. In this specific case the difference between the two models is not marked, but in most cases one model represents the data better than the other.

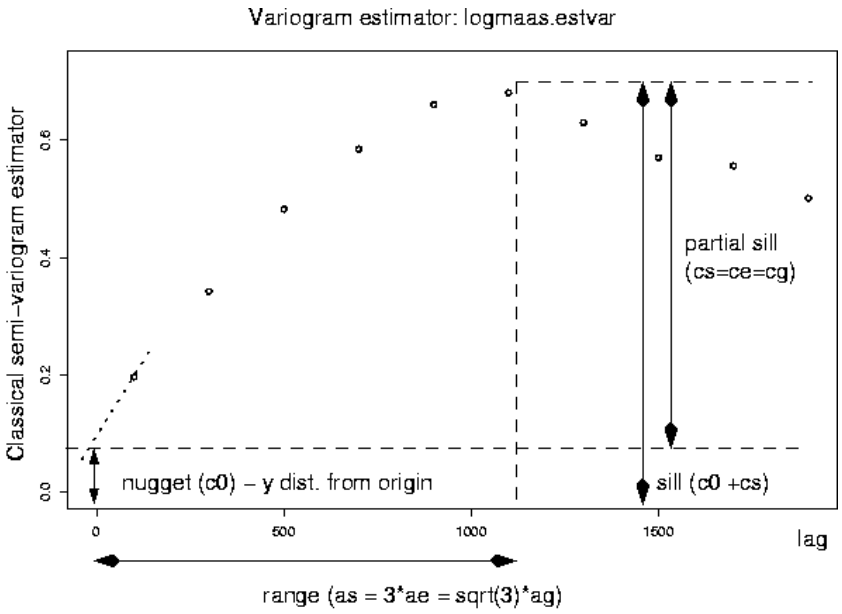

Figure 10.18: Schematic diagram for the estimate of the coefficients to input in the semivariogram model.

By inputting the output map name and confirming with *ok* the actual interpolation process begins. There are two output maps, one of the interpolated values on a regular grid and the other with an estimate of the error, also referred to the regular grid. The estimate of the error is very useful for the validation of the interpolation process, the closer the error value is to the interpolated value the less significant the interpolated value is and, therefore, the less valid the interpolation is as a whole. Of course, the estimate of the error is greater the further you are from a measured value. At the end of the procedure, a message will inform you of the creation of the two maps.

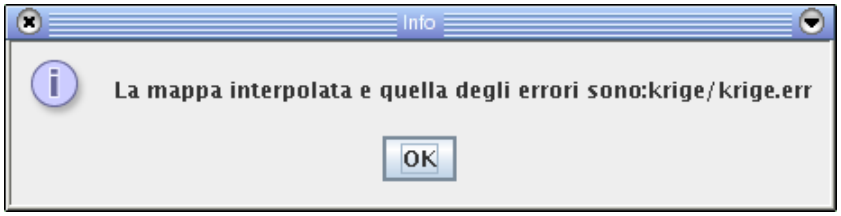

Figure 10.19: Final message of the interpolation procedure with the details of the two maps that were created.

If the selection of the variogram model were not correct, or the parameters chosen were not optimal, the display would show either a vertical or a horizontal line (figure 10.20). The resulting semivariogram is clearly unusable in the kriging interpolation.

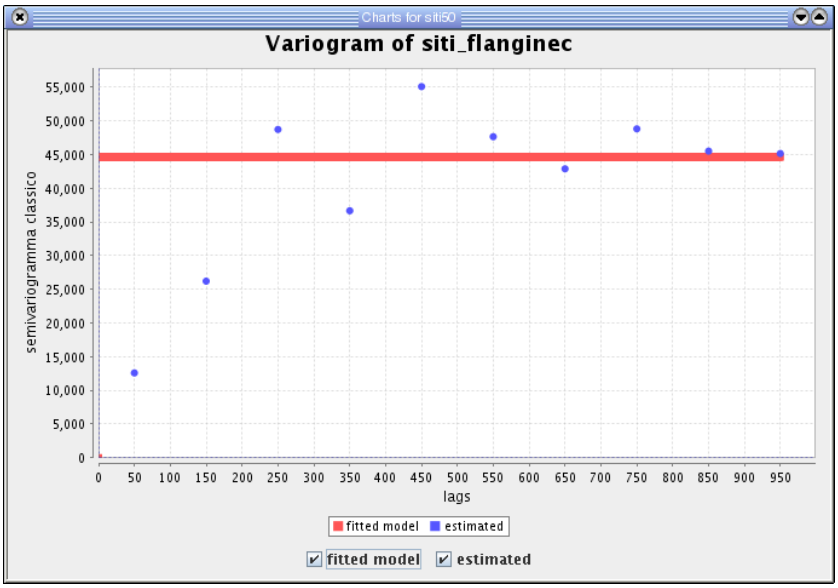

Figure 10.20: Graph with the superimposition of the real semivariogram and one obtained with the exponential model and wrong parameters.

The two maps will contain valid values in the entire active data region, to trim them down to a surface smaller than the active region just use the *r.mapcalc* command.

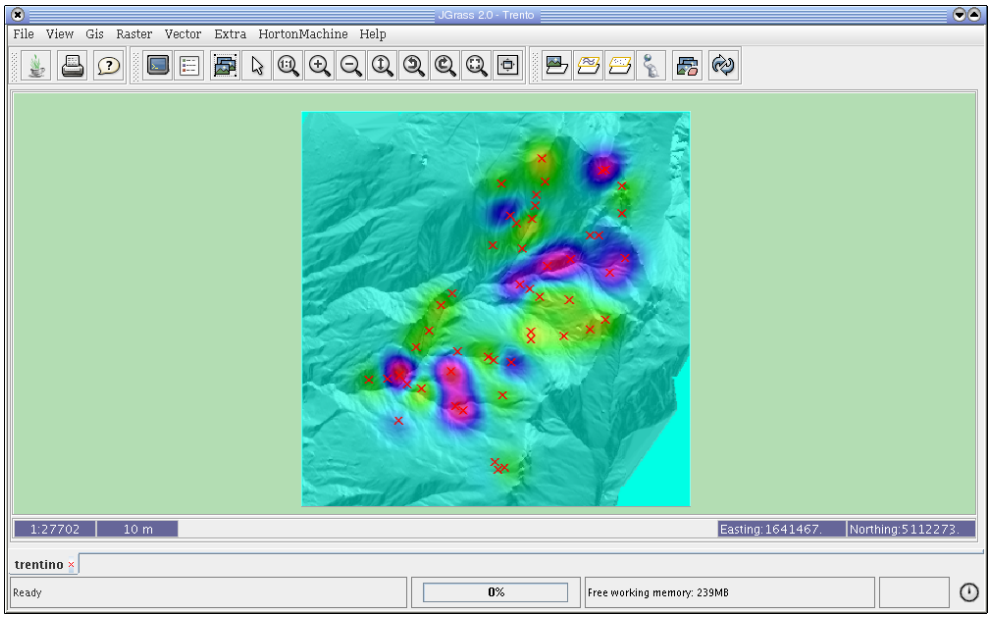

Figure 10.21: The point map interpolated by kriging.

11 Printing

Printing has always been one of the main problems of Open Source GIS. Printing the result of the various operations is of fundamental importance in divulging and presenting results.

In *JGrass*, a handy print editor for the printing of maps that are displayed on screen, on various paper formats, is in its development stages. It is possible to prepare a print with a predefined template, or draw and insert new objects and text.

Before describing the operations that must be carried out during the printing phase it is important to clarify that at the state of the art the printing module is not yet optimized. This does not mean that the printing feature does not work or that it is totally unusable, but it advisable to print only in A4 format, or at the most in A3, because "large" prints of the maps are not of good quality. The part of the module which is in development is expected to redraw the map at the scale of the sheet of paper so as to always have the greatest resolution possible for the print. Unfortunately, as of yet, it has not been possible to develop the algorithm completely.

To print you only need to click on the icon and the print editor will open (figure 11.1). The editor is split into two parts, on the left there are all the print options, on the right the print layout is displayed.

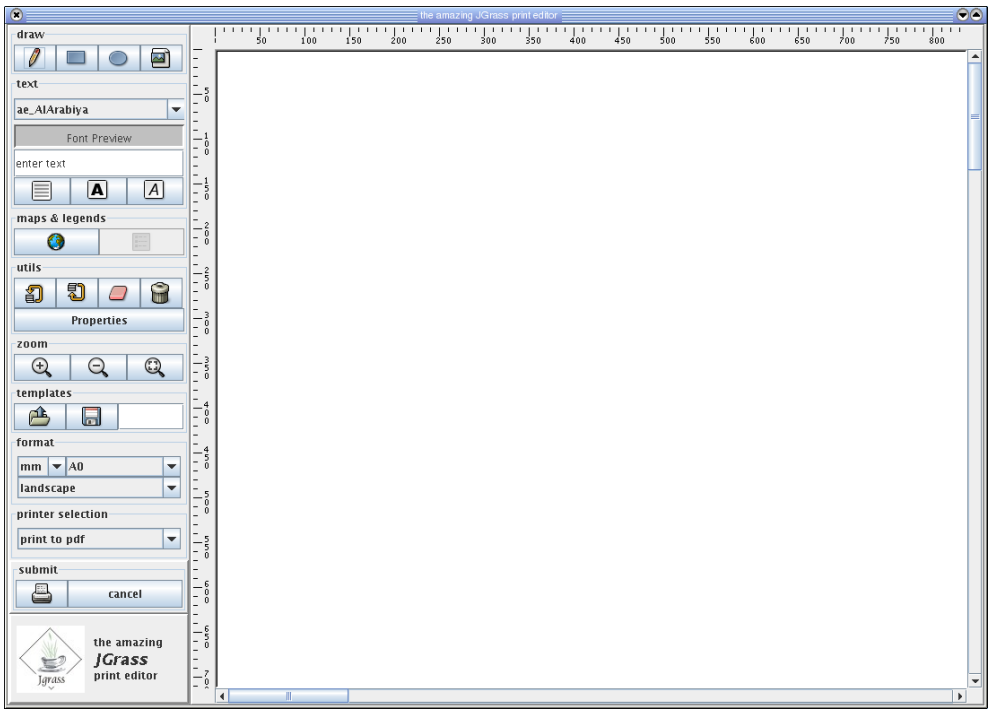

Figure 11.1: The *JGrass* print editor.

Among the print options there are:

- **draw**: to insert drawings in the print layout such as lines, rectangles, external images (which can be used in the cartouches of project plans, for example);
- **text**: to define the font type and the text settings;
- **maps and legends**: to insert the displayed maps and the respective legends (this option is not yet active);
- **utils**: to edit the settings of the drawn objects (to manage the superimpositions and remove one or more drawn objects);
- **zoom**: to zoom in to the layout sheet for the definition of the objects;
- **templates**: to load and save print templates (Paragraph 4.3);
- **format**: to choose the the paper format and orientation;
- **printer selection**: at the moment it only allows to print to pdf file;
- **submit**: to launch the print process or cancel the work done.

To insert an object in the print layout you only have to select the object, or image, from the menu and then use the mouse to select the area the object should cover in the print layout. The term *object* refers to objects such as lines and polygons, as well as to displayed text and maps .

An example of the print of a map with its respective legend, taken from the *JGrass* display window, is shown in figure 11.2.

Warning: for certain combinations of operating system and video card the visualization of the map in the layout sheet may not work, giving a completely black window.

The printing of maps in *JGrass* is done at the same scale as the on-screen display. You need to set the display scale which will then be used in the prints.

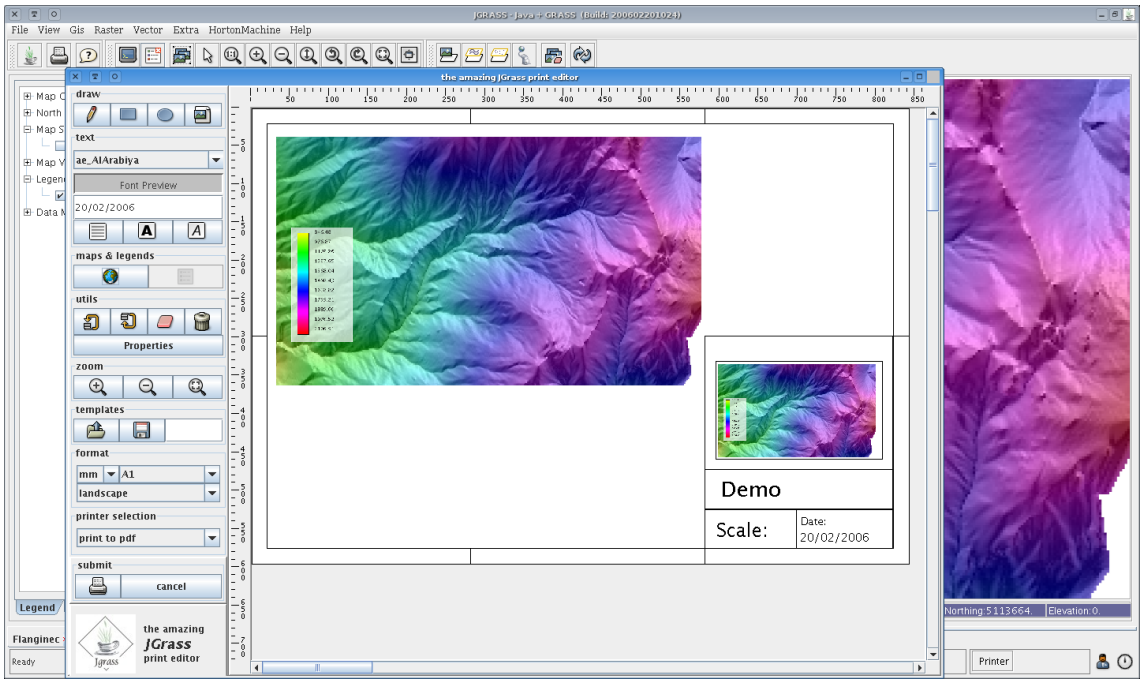

Figure 11.2: The *JGrass* print editor.

12 Uninstalling

To uninstall the program you only need to click the uninstall icon and the uninstall wizard will launch, which also gives you the possibility to remove the *JGrass* directory from the filesystem.

Another uninstall option is simply to remove the directory where *JGrass* was installed. In fact, *JGrass* is completely installed in its installation directory and does not need files in other filesystem directories.

The more expert users can also uninstall the program from the console command line by executing the uninstall jar file found in the *Uninstaller* directory of the *JGrass* installation directory.